Data Aggregation and Demand Prediction

Maxime C. Cohen
Desautels Faculty of Management, McGill University, maxime.cohen@mcgill.ca

Renyu Zhang
NYU Shanghai and the Chinese University of Hong Kong, renyu.zhang@nyu.edu and philipzhang@cuhk.edu.hk

Kevin Jiao
NYU Stern School of Business, New York, jjiao@stern.nyu.edu

We study how retailers can use data aggregation and clustering to improve demand prediction. High accuracy in demand prediction allows retailers to effectively manage their inventory as well as mitigate stock-outs and excess supply. A typical retail setting involves predicting demand for hundreds of items simultaneously. Although some items have a large amount of historical data, others were recently introduced and, thus, transaction data can be scarce. A common approach is to cluster several items and estimate a joint model for each cluster. In this vein, one can estimate some model parameters by aggregating the data from several items and other parameters at the individual-item level. We propose a practical method referred to as *Data Aggregation with Clustering* (DAC), which balances the trade-off between data aggregation and model flexibility. DAC allows us to predict demand while optimally identifying the features that should be estimated at the (i) item, (ii) cluster, and (iii) aggregate levels. We show that the DAC algorithm yields a consistent and normal estimate, along with improved prediction errors relative to the decentralized benchmark, which estimates a different model for each item. Using both simulated and real data, we illustrate DAC's improvement in prediction accuracy relative to a wide range of common benchmarks. Interestingly, the DAC algorithm has theoretical and practical advantages and helps retailers uncover meaningful managerial insights.

Key words: Retail analytics, demand prediction, data aggregation, clustering

1. Introduction

Retailers routinely collect large volumes of historical data, which are used to improve future business practices, such as inventory management, pricing decisions, and customer segmentation. One of the most important data-driven tasks for retailers is to predict the demand for each stock-keeping unit (SKU). A common approach in practice is to classify SKUs into different departments (e.g., soft drinks) and sometimes even into sub-categories (e.g., a specific types of soft drinks) and then build predictive models accordingly. A typical demand prediction model is a regression specification with the sales (or logarithmic of the sales) as the outcome variable and price, seasonality, brand, color, and promotions as features. The model coefficients are then estimated using historical data.

In many retail settings, a subset of items has been offered for a long time (referred to as 'old items'), whereas other items were recently introduced. While the demand prediction for old items is generally

easy due to abundant data availability, accurately predicting the demand for newly introduced items with a limited number of historical observations is considerably more challenging. One may then wonder how the available data of old items from the same department could be leveraged to enhance the prediction of new items. Indeed, SKUs in the same department often share similar characteristics and, hence, tend to be affected by a particular feature in a similar way. A prominent approach is to estimate certain coefficients at an aggregate level (i.e., by gathering the data across all SKUs and assuming a uniform coefficient). For example, it seems reasonable to believe that all items in the ice-cream category share the same seasonal patterns. Although this approach has been widely adopted in the retail industry, no rigorous empirical method has been developed to formalize how this data aggregation procedure should be applied for demand prediction. In this paper, we seek to bridge this gap by formalizing the tradeoff between data aggregation (i.e., pooling data from different items to reduce variance) and model flexibility (i.e., estimating a different model for each item to reduce bias) in a systematic fashion.

Due to insufficient data, the traditional approach of estimating a different model for each SKU is usually inefficient for new products or SKUs with noisy observations. This approach cannot identify the right aggregation level for each coefficient and does not find the underlying cluster structure of the coefficients. Based on common clustering methods (e.g., k-means), we propose an efficient and integrated approach to infer the coefficient of each feature while identifying the right level of data aggregation based on the statistical properties of the estimated coefficients. Our method also allows us to incorporate multiple aggregation levels while preserving model interpretability. From a theoretical perspective, our method yields a consistent estimate, along with improved asymptotic properties. From a practical perspective, our method can easily be estimated using retail data and significantly improves out-of-sample prediction accuracy.

1.1. Main Results and Contributions

We study the trade-off between data aggregation and model flexibility by optimally identifying the right level of aggregation for each feature, as well as the cluster structure of the items. We propose a practical method—referred to as the Data Aggregation with Clustering (DAC) algorithm, which allows us to predict demand while optimally identifying the features that should be estimated at the (i) item, (ii) cluster, and (iii) aggregate levels. Our proposed algorithm first applies maximum-likelihood estimation to obtain a different coefficient vector for each item (called the decentralized model). It then performs a hypothesis test (i.e., t-test) on the estimated coefficients from the decentralized model to identify the correct aggregation level for each feature. To characterize the cluster structure of the items, we apply the k-means method on the estimated coefficients from the decentralized model (as opposed to using features' average values).

We first characterize the DAC algorithm's theoretical properties. Specifically, we show that it yields a consistent estimate of the data aggregation levels and cluster structures. The estimated feature coefficients under DAC are consistent and normal. Thus, if the data has enough observations, one can correctly identify the underlying data generating process. In addition to this consistency and normality result, we derive improved prediction errors—variance, mean squared error, and generalization error are all smaller—relative to the commonly-used MLE method applied in a decentralized fashion to each item. Furthermore, we show that if some items have abundant data while other items have limited data, our proposed algorithm improves the prediction accuracy for all items, with a more significant improvement for the items with limited data. Armed with these theoretical results, we then conduct extensive computational experiments based on both simulated and real data to illustrate the DAC algorithm's significantly improved prediction accuracy relative to 15 different benchmarks. Our results highlight the essential value of the DAC algorithm in better balancing the bias-variance trade-off, resulting in more accurate demand prediction. Furthermore, our algorithm can accurately identify the right data aggregation levels and recover cluster structure in the non-asymptotic regime (i.e., when the sample size of the data set is finite). Finally, we apply the DAC algorithm using two years of retail data and find that it can also help retailers uncover useful insights on the relationships between the different items.

1.2. Related Literature

This paper is related to several literature streams, including prediction and clustering algorithms, retail operations, and demand forecasting.

Prediction and clustering algorithms: The problems of demand prediction and clustering have been extensively studied in the machine learning (ML) literature. Donti et al. (2017) focus on developing new ML methods by training a prediction model to solve a nominal optimization problem. Although several studies have focused on general settings, it is difficult to apply existing methods to a retail setting where multiple levels of hierarchy may exist. Elmachtoub and Grigas (2017) propose a new idea called 'smart predict, then optimize' (SPO). SPO's key feature is that the loss function is computed based on comparing objective values generated by using predicted and observed data. The authors then address the computational challenge and develop a tractable SPO version. Jagabathula et al. (2018) propose a model-based embedding technique to segment a large population of customers into non-overlapping clusters with similar preferences. Bertsimas and Kallus (2020) combine ideas from ML and operations research to propose a new prediction method. The authors solve a conditional stochastic optimization problem by incorporating various ML methods, such as local regression and random forests. Liu et al. (2021) apply clustering techniques to predict the travel time of last-mile delivery services and optimize the order assignment. Our work is also related

to the traditional clustering literature. Since the introduction of k-means by MacQueen et al. (1967), clustering algorithms have been extensively studied. In the context of assortment personalization, Bernstein et al. (2019) propose a dynamic clustering method to estimate customer preferences. In our paper, we leverage some theoretical properties of the k-means clustering method and embed it as one of the key steps in our demand prediction algorithm.

Retail operations and demand forecasting: Retailers are always seeking ways to improve operational decisions, such as inventory replenishment, supply chain management, and pricing. These decisions rely heavily on accurate demand prediction. We refer interested readers to Fildes et al. (2019b) for a comprehensive review of this literature, which demonstrates that forecasters in retailing often face the dimensionality problem of having too many features but too little data. As reported by Cohen and Lee (2020), demand uncertainty is a major issue in designing efficient global supply chains. There is an extensive body of literature focused on developing methods for demand prediction in retail settings. Sophisticated models have been developed in the past two decades to manage the increasing volume of data collected by retailers. Marketing papers, such as Van Heerde et al. (2000) and Macé and Neslin (2004), study pre- and post-promotion dips using linear regression models with lagged variables. Kök and Fisher (2007) develop a procedure to estimate substitution behavior in retail demand. Recent developments in demand prediction include the following three papers: Huang et al. (2014), who embed competitive information (including price and promotions) into demand prediction; Fildes et al. (2019a), who suggest that promotional information can be quite valuable in improving forecast accuracy; and Huang et al. (2019), who further account for the impact of marketing activities. Ma et al. (2016) develop a Lasso-based four-step methodological framework to overcome the problem of the ultra-high dimensionality of the feature space under multiple product categories. In the operations management community, demand prediction models are often used as an input to an optimization problem (e.g., Cohen et al. 2017, 2021). Specifically, Cohen et al. (2017) estimate a log-log demand model using supermarket data. The authors then solve the promotion optimization problem by developing an approximation based on linear programming. It was shown in the retail operations literature that responding to accurate demand forecasts can substantially increase profits (Caro and Gallien 2010). Kesavan et al. (2010) show that incorporating the cost of goods sold, inventory, and gross margin information can substantially improve sales forecasting for retailers. In recent years, the amount of data available has grown exponentially, thus offering new opportunities for research on demand prediction (Feng and Shanthikumar 2018). In this context, our paper proposes a new demand prediction method that can efficiently aggregate data from multiple items to improve prediction accuracy.

When the demand data show too high variation or is insufficient to construct reliable forecasting models, appropriately pooling data to improve the prediction accuracy has been widely studied in

the demand forecasting literature. For example, Cooper et al. (1999) and Cooper and Giuffrida (2000) propose pooled regression models and residuals to extract information and draw managerial insights on the impact of retail promotions. Dekker et al. (2004) present forecasting methods that pool demand information from a higher aggregation level and smartly combines forecasts thereof. In a time-series framework, Gür Ali et al. (2009) find the value of pooling observations from different stores to improve demand prediction accuracy. By pooling information from different stores and SKUs, Gür Ali (2013) propose a "Driver Moderator" that produces short-term forecasts for both existing and new SKUs. In an online fashion setting, Ferreira et al. (2016) propose non-parametric machine learning techniques to aggregate data from existing SKUs for demand prediction and price optimization of new SKUs that have never been sold before. Our main contribution in this literature is that we provide a systematic framework and an associated DAC algorithm to efficiently pool data, along with rigorous theoretical justifications and the edge of our approach over the decentralized benchmark (both theoretically and in practice), whereas most existing approaches in this literature are of heuristic nature and bear no theoretical performance guarantee or validation.

A recent stream of papers integrate a clustering step into demand prediction. For instance, Baardman et al. (2017) propose an iterative approach to cluster products and leverage existing data to predict the sales of new products. Hu et al. (2019) propose a two-step approach to first estimate the product lifecycle and then cluster and predict. Park et al. (2017) develop a generalized clusterwise linear regression model and propose algorithms to predict the demand of multiple SKUs. In our paper, however, the definition of clusters is fundamentally different. Unlike previous studies, our clustering is based on the estimated coefficients rather than the features' values. Furthermore, our model is flexible enough to account for different levels of data aggregation, whereas in previous studies, all features are essentially estimated at the cluster level. Allowing such flexibility is key to improving demand forecasting.

Structure of the paper. The remainder of the paper is organized as follows. In Section 2, we introduce our model and discuss the relevant computational challenges. We then describe the DAC algorithm in Section 3. Our analytical results on the value of our proposed algorithm are presented in Section 4. In Sections 5 and 6, we conduct computational experiments using simulated and real data, respectively. Our conclusions are reported in Section 7. The proofs of our analytical results are relegated to Appendix C.

2. Model

We introduce our demand prediction model under the generalized linear model (GLM) framework. Specifically, we consider a retail department (e.g., soft drinks or electronics) comprising n items (or SKUs). Each item has m historical observations (e.g., weekly sales transactions). We will show

in Section 3 that our model and method can be straightforwardly generalized to a setting where different items have a different number of observations. For item i and observation j ($1 \le i \le n$ and $1 \le j \le m$), we denote the (log-of-)sales as $Y_{i,j}$ and the feature vector (e.g., price, promotion status, seasonality, functionality, and color) as $\boldsymbol{X}_{i,j} := (X^1_{i,j}, X^2_{i,j}, \ldots, X^d_{i,j})' \in \mathbb{R}^d$, which is i.i.d. with respect to observation j and independent with respect to item i. Without loss of generality, we assume that $\boldsymbol{X}_{i,j}$ has a bounded support with $\mathbb{E}[\boldsymbol{X}_{i,j}] = \mathbf{0} \in \mathbb{R}^d$ and a positive definite second-moment matrix $\Sigma_i := \mathbb{E}\left[\boldsymbol{X}_{i,j}\boldsymbol{X}'_{i,j}\right]$ for each item i, that is, the smallest eigenvalue $\lambda_{\min}(\Sigma_i) > 0$. This is a standard assumption to ensure identification and consistency in the statistics literature (e.g., Fahrmeir et al. 1985). The feature set is denoted by $\mathcal{D} := \{1, 2, \ldots, d\}$.

An important characteristic of our model is that a feature $l \in \mathcal{D}$ may affect the demand of an item at different data aggregation levels: (i) SKU, (ii) cluster, and (iii) department. More precisely, a feature may have the same impact on all items, captured by a uniform coefficient for all items in the department. We refer to such features as *shared* (i.e., department-level features), the set of which is denoted by $\mathcal{D}_s$. Here, we consider a setting where all the items belong to the same department to be consistent with the usual business retail practice, where demand prediction is often performed for each department separately. We highlight, however, that our approach can be directly applied to a more general setting without a department structure. Alternatively, a feature may have a different impact on different items, captured by a different coefficient for each item. We refer to such features as *non-shared* (i.e., SKU-level features), the set of which is denoted by $\mathcal{D}_n$. Finally, we assume that the items are segmented into different clusters so that some features have the same impact on items within the same cluster and a different impact on items from a different cluster. This phenomenon is captured by a uniform coefficient for all items in the same cluster (the coefficients are different for each cluster). We refer to such features as cluster-level features, the set of which is denoted by $\mathcal{D}_c$. Without loss of generality, we assume that, for each cluster-level feature l, the number of clusters k_l or the way that the clusters are formed may be different. Thus, the entire feature set, $\mathcal{D}$, can be written as the union of three disjoint sets of features that affect the demand at different aggregation levels: $\mathcal{D} = \mathcal{D}_s \cup \mathcal{D}_n \cup \mathcal{D}_c$. The aggregation structure $\mathcal{D}_s$, $\mathcal{D}_n$, and $\mathcal{D}_c$, and the cluster partition of the items are unknown a priori and will be estimated from data.

We assume that the ground truth follows the GLM specification. Specifically, the observations are generated from an exponential family distribution that includes normal, binomial, gamma, Poisson, and inverse-normal distributions as special cases. We refer to Fahrmeir et al. (1985) and McCullagh and Nelder (2019) for an introduction of the standard theory of GLM. Based on the three data aggregation levels of the features, we have

$$Y_{i,j} = G\Big(\sum_{l\in\mathcal{D}_s} X^l_{i,j}\beta^s_l + \sum_{l\in\mathcal{D}_n} X^l_{i,j}\beta^n_{i,l} + \sum_{l\in\mathcal{D}_c} X^l_{i,j}\beta^c_{\varsigma(i,l),l}\Big) + \epsilon_{i,j}, \qquad i = 1,\ldots,n \text{ and } j = 1,\ldots,m. \quad (1)$$

Here, $\varsigma(i,l) \in \{1,2,\ldots,k_l\}$ is the cluster index that contains item i with respect to feature l, $G(\cdot)$ represents the strictly increasing link function that establishes the relationship between the linear predictor and the mean of the outcome variable, and $\epsilon_{i,j}$'s are independent zero-mean random noises. We use $\mathcal{C}_{\varsigma,l} \subset \{1,2,\ldots,n\}$ to denote the cluster with index ς with respect to feature l, where $\varsigma \in \{1,2,\ldots,k_l\}$ and $\{\mathcal{C}_{1,l},\mathcal{C}_{2,l},\ldots,\mathcal{C}_{k_l,l}\}$ form a partition of the items $\{1,2,\ldots,n\}$. We also call $\{\mathcal{C}_{1,l},\mathcal{C}_{2,l},\ldots,\mathcal{C}_{k_l,l}\}$ the cluster structure with respect to feature l. We denote $\mathcal{C}(i,l) := \mathcal{C}_{\varsigma(i,l),l}$ as the cluster that contains item i with respect to feature $l \in \mathcal{D}_c$. We also define $\mathcal{C}(i,l) := \{i\}$ if $l \in \mathcal{D}_n$, and $\mathcal{C}(i,l) := \{1,2,...,n\}$ if $l \in \mathcal{D}_s$. There are many commonly-used link functions, and in practice, the function depends on the context. For example, if $Y_{i,j}$ is the number of sold units of item i in observation j, $G(u) = u$ can be the identity function and, thus, the model reduces to a linear regression. On the other hand, if $Y_{i,j}$ is a binary variable, $G(u) = 1/(1+\exp(-u))$ can be the sigmoid function. Likewise, there exist other examples of link functions, such as logarithmic and inverse squared. We assume that $\epsilon_{i,j}$ is sub-Gaussian with parameter $\sigma > 0$, i.e., $\mathbb{E}[\exp(\lambda\epsilon_{i,j})] \le \exp(\lambda^2\sigma^2/2)$ for any λ, which is a standard assumption in the statistics and ML literature. We assume that all the observations are *independent* across both time periods and items. Extensions of the model and algorithm via vector auto-regression (resp. generalized least squares) to cases where observations may be correlated across time periods (resp. items) are discussed at the end of Section 3. We also define $\beta_{i,l}$ as the coefficient of $X^l_{i,j}$ in the GLM specification in Eq. (1), i.e., $\beta_{i,l} = \beta^s_l$ if $l \in \mathcal{D}_s$, $\beta_{i,l} = \beta^n_{i,l}$ if $l \in \mathcal{D}_n$ and $\beta_{i,l} = \beta^c_{\varsigma(i,l),l}$ if $l \in \mathcal{D}_c$. We denote $\boldsymbol{\beta}_i := (\beta_{i,1},\beta_{i,2},...,\beta_{i,d})'$ as the *true* coefficient vector for item i and $\boldsymbol{\beta} := (\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,...,\boldsymbol{\beta}_n)$ as the *true* coefficient matrix for all items. Likewise, we use $\hat{\boldsymbol{\beta}} := (\hat{\beta}_{i,l} : 1 \le i \le n, 1 \le j \le d)$ to denote an estimator of $\boldsymbol{\beta}$. So $\hat{\beta}_{i,l}$ is the estimator for $\beta_{i,l}$.

Based on the GLM specification in Eq. (1), we can characterize the aggregation levels of the three types of features. For a department-level feature $l \in \mathcal{D}_s$, its coefficient β^s_l is shared among *all* items. Thus, all items in the department will have the same coefficient for this feature. In comparison, for an SKU-level feature $l \in \mathcal{D}_n$, its coefficient $\beta^n_{i,l}$ will differ across items (i.e., $\beta^n_{i,l} \ne \beta^n_{i',l}$ for $i \ne i'$). Finally, for a cluster-level feature $l \in \mathcal{D}_c$, all items in the same cluster will have the same coefficient (i.e., for all $i \ne i'$ with $\varsigma(i,l) = \varsigma(i',l) = \varsigma$, the coefficient of $X^l_{i,j}$ and that of $X^l_{i',j}$ are both $\beta^c_{\varsigma,l}$). Thus, the total number of coefficients for all n items is $d_x := n|\mathcal{D}_n| + \sum_{l\in\mathcal{D}_c} k_l + |\mathcal{D}_s| \le nd$. We use $n_{i,l}$ to denote the number of items that share the same coefficient as item i for feature l (i.e., $n_{i,l} = 1$ if $l \in \mathcal{D}_n$, $n_{i,l} = n$ if $l \in \mathcal{D}_s$, and $n_{i,l} = |\mathcal{C}(i,l)|$ if $l \in \mathcal{D}_c$). Estimating the coefficient of some features at a certain level of aggregation is common in practice. For example, retailers often estimate seasonality coefficients at the department level to avoid overfitting and capture the fact that the items follow the same seasonal patterns. Furthermore, when estimating the effect of promotions on demand, such as cannibalization and halo effects, one may cluster several items together because promotions often have a similar impact on a group of items. Assumption 1 below simplifies the exposition by avoiding

the situation where two clusters have the same coefficient value. Without loss of generality, we make the following assumption throughout the paper for expositional and computational convenience.

ASSUMPTION 1. *(a) If a feature l is at the SKU level (i.e., $l \in \mathcal{D}_n$), then $\beta_{i,l} \neq \beta_{i',l}$ for any pair of items i and i', that is, a SKU-level feature has a different effect on each item.*

(b) For $l \in \mathcal{D}_c$, we have $\beta_{i,l} = \beta_{i',l}$ if and only if $\mathcal{C}(i,l) = \mathcal{C}(i',l)$, that is, a cluster-level feature has the same effect on items in the same cluster and a different effect for different clusters. Furthermore, each cluster has at least two items.

Our main goal is to accurately predict the outcome variable Y (in our case, weekly demand) given the feature vector $\boldsymbol{X}$ (e.g., price, promotion, seasonality, or color), assuming that the data generating process follows Eq. (1). Before presenting our proposed demand prediction method, we first discuss the key challenge of fitting the model by using two intuitive methods. First, one could adopt the idea of lasso regression (see, e.g., Tibshirani 1996, Tibshirani and Taylor 2011, Hastie et al. 2019) and estimate the coefficients through the *generalized ℓ_1-regularized maximum-likelihood estimator* (MLE). This approach revises the standard MLE by adding a generalized ℓ_1-regularizer

$$-\lambda\Big(\sum_{i\neq i'}\sum_{l=1}^{d}|\beta_{i,l}-\beta_{i',l}|\Big)$$

to the log-likelihood function (see Eq. (19) in Appendix B.1). A canonical result in the statistics literature shows that ℓ_1-regularization will shrink the regularized terms to 0 and, thus, generate sparse solutions. As a result, adding a generalized ℓ_1-regularizer to MLE may potentially be helpful to capture the fact that a feature at the aggregate or cluster level shares the same coefficient for different items. We note that this approach is in a similar spirit to the *fused lasso regression* (see, e.g., Tibshirani and Taylor 2011). Given the high-dimensional nature of the regularized MLE problem in Eq. (19) (i.e., the number of decision variables is nd, which is at the magnitude of thousand or more in practice), estimating the coefficients is computationally prohibitive even for a linear regression specification (i.e., $G(u) = u$) as it involves inverting $(nd) \times (nd)$-matrices in each step to construct the solution path. We provide a more detailed discussion on this approach applied to our problem in Appendix B.1.

A second possible approach is via *direct optimization*, which is based on explicitly estimating the feature aggregation levels and cluster structures. This approach introduces one-hot encoding binary variables to represent the aggregation level of each feature and the cluster that each item belongs to. To simultaneously estimate (i) the aggregation level of each feature, (ii) the cluster that each item belongs to, and (iii) the coefficient of each feature for each item, one may either reformulate the optimization as a mixed-integer second-order conic program (SOCP), as in Eq. (28) or iteratively

estimate the aggregation level, cluster structure, and the coefficients by varying one of these three types of decisions and fixing the other two to maximize the maximum likelihood. In a practical setting, however, the second-order SOCP will have a too-high dimension to be computationally tractable. The iterative procedure will stop once the binary variables remain the same for two consecutive iterations. A similar iterative optimization approach was proposed by Baardman et al. (2017) to address the demand prediction problem with only cluster-level features (i.e., no department-level and no SKU-level features). In their setting, this iterative procedure was proved to converge to the true coefficients and cluster structure (i.e., the estimate is consistent). In our setting, however, the consistency of the approach proposed by Baardman et al. (2017) is not guaranteed. See Appendix B.2 for more details on the direct optimization approach.

3. Data Aggregation with Clustering

As mentioned, simultaneously estimating the aggregation levels, cluster structures, and feature coefficients is computationally challenging and prone to substantial prediction errors. In this section, we propose a novel approach that allows us to (a) identify the correct level of aggregation for each feature, (b) find the underlying cluster structure of the SKUs with respect to each feature, and (c) generate a consistent estimate of the feature coefficients. Our method is entirely data-driven and can efficiently achieve these three goals in an integrated fashion while yielding an accurate demand prediction.

We begin our analysis by focusing on a (simple) special case of the GLM in Eq. (1), where all the features are at the SKU level. In this case, the data-generating process can be written as

$$Y_{i,j} = G\Big(\sum_{l\in\mathcal{D}} X^l_{i,j} b_{i,l}\Big) + \epsilon_{i,j}, \qquad i = 1,\dots,n \text{ and } j = 1,\dots,m. \tag{2}$$

By comparing the model specifications in (1) and (2), we have $b_{i,l} = \beta^s_l$ for $l \in \mathcal{D}_s$, $b_{i,l} = \beta^n_{i,l}$ for $l \in \mathcal{D}_n$, and $b_{i,l} = \beta^c_{\varsigma(i,l),l}$ for $l \in \mathcal{D}_c$. We refer to model (2) as the *decentralized model* since each item is fitted in a decentralized fashion. Estimating the decentralized model is usually carried out through iterative re-weighted least squares, which ultimately lead to MLE (see Appendix A and McCullagh and Nelder 2019, for more details). We assume that for each item, the decentralized model is well defined with a unique MLE solution, which is the case for commonly used GLMs, such as linear and logistic regression. Estimating the decentralized model can be decomposed into estimating one model for each item separately. Using the data of item i, we apply the MLE to find the estimated coefficients of this item, $\hat{\boldsymbol{b}}_i := (\hat{b}_{i,1}, \hat{b}_{i,2}, \dots, \hat{b}_{i,d})' \in \mathbb{R}^d$ as follows:

$$\hat{\boldsymbol{b}}_i \in \arg\max_{\boldsymbol{b}_i} \sum_{j=1}^{m} \log \mathcal{L}_i(\boldsymbol{b}_i | Y_{i,j}, \boldsymbol{X}_{i,j}) = \arg\max_{\boldsymbol{b}_i} \sum_{j=1}^{m} \left[Y_{i,j} \boldsymbol{X}'_{i,j} \boldsymbol{b}_i - H(\boldsymbol{X}'_{i,j} \boldsymbol{b}_i) \right], \tag{3}$$

where $\mathcal{L}_i(\boldsymbol{b}_i|Y_{i,j},\boldsymbol{X}_{i,j})$ is the likelihood function associated with the data $(Y_{i,j},\boldsymbol{X}_{i,j})$ and the coefficient vector $\boldsymbol{b}_i\in\mathbb{R}^d$, and $H(\cdot):\mathbb{R}\to\mathbb{R}$ is the infinitely differentiable normalization mapping in the GLM with $H'(u)=G(u)$ (see Appendix A for details). We refer to the estimator $\hat{\boldsymbol{b}}:=(\hat{\boldsymbol{b}}_1,\hat{\boldsymbol{b}}_2,\ldots,\hat{\boldsymbol{b}}_n)$ as the *decentralized estimator*. Throughout this paper, we parameterize the estimators with the sample size m when we want to make this dependence explicit. For example, we use $\hat{\boldsymbol{b}}(m):=(\hat{\boldsymbol{b}}_1(m),\hat{\boldsymbol{b}}_2(m),\ldots,\hat{\boldsymbol{b}}_n(m))$ to denote a decentralized estimator with sample size m. Also, we define the Fisher information matrix with respect to the decentralized model of item i as

$$\mathcal{I}_i(\boldsymbol{b}_i):=-\mathbb{E}[\nabla_2\log\mathcal{L}_i(\boldsymbol{b}_i|Y_{i,j},\boldsymbol{X}_{i,j})],$$

where ∇_2 is the Hessian operator and the expectation is taken with respect to $(Y_{i,j},\boldsymbol{X}_{i,j})$. We first show the following consistency and normality property of the decentralized estimator $\hat{\boldsymbol{b}}$, which will be used as a building block for our subsequent analyses.

PROPOSITION 1. *The decentralized estimator $\hat{\boldsymbol{b}}$ satisfies the following properties:*

(a) CONSISTENCY. *As $m\uparrow+\infty$, (i) if $l\in\mathcal{D}_s$, then $\hat{b}_{i,l}(m)\xrightarrow{p}\beta^s_l$; (ii) if $l\in\mathcal{D}_n$, then $\hat{b}_{i,l}(m)\xrightarrow{p}\beta^n_{i,l}$; and (iii) if $l\in\mathcal{D}_c$, then $\hat{b}_{i,l}(m)\xrightarrow{p}\beta^c_{\varsigma(i,l),l}$, where $\xrightarrow{p}$ refers to convergence in probability.*

(b) NORMALITY. *For each item i and each feature $l\in\mathcal{D}$, there exist a threshold $\mathfrak{m}_{i,l}$ on the sample size and a constant $\psi_{i,l}>0$ such that if $m\geq\mathfrak{m}_{i,l}$, we have, for any $\epsilon>0$,*

$$\mathbb{P}(|\hat{b}_{i,l}(m)-\beta_{i,l}|\leq\epsilon)\geq 1-3\exp(-\psi_{i,l}\epsilon^2 m). \tag{4}$$

Furthermore, for each item i, $\hat{\boldsymbol{b}}_i$ is asymptotically normally distributed with

$$\sqrt{m}(\hat{\boldsymbol{b}}_i(m)-\boldsymbol{\beta}_i)\xrightarrow{d}\mathcal{N}(\mathbf{0},\mathcal{I}_i(\boldsymbol{\beta}_i)^{-1})\ \text{ as } m\uparrow+\infty, \tag{5}$$

where $\xrightarrow{d}$ refers to convergence in distribution, $\mathcal{N}(\mathbf{0},\mathcal{I}_i(\boldsymbol{\beta}_i)^{-1})$ refers to the multivariate normal distribution with mean $\mathbf{0}\in\mathbb{R}^d$ and covariance matrix $\mathcal{I}_i(\boldsymbol{\beta}_i)^{-1}$.

Proposition 1(a) shows that with sufficiently many observations, we can consistently estimate the feature coefficients using the decentralized model. It is unsurprising that the decentralized estimator, $\hat{\boldsymbol{b}}$, is consistent given that the decentralized model has a high amount of flexibility (which implies a low bias; see also Fahrmeir et al. 1985). Proposition 1(b) further demonstrates that the coefficient estimation error of the decentralized approach is approximately normally distributed under both finite sample and asymptotic regimes. Therefore, if we have sufficiently many observations for each item, the prediction accuracy of the decentralized model will be high. However, two issues remain unaddressed with the decentralized estimation: (i) how can we find the right aggregation level for each feature, and (ii) how can we identify the items' cluster structure. Furthermore, the decentralized

estimation may suffer from overfitting and, hence, admit a high variance. Addressing these issues will be the main focus in the remaining of this paper.

To estimate the aggregation level and the underlying cluster structure based on the GLM specification in Eq. (1), we next introduce another special case of the model in which the data aggregation level and the cluster structure are known to the retailer. We refer to this case as the *aggregate model* and call its MLE the *aggregate estimator*, which we denote by

$$\hat{\boldsymbol{b}}^a \in \arg\max_{\boldsymbol{\beta}\in\Xi} \sum_{j=1}^{m}\sum_{i=1}^{n} \log \mathcal{L}(\boldsymbol{\beta}_i|Y_{i,j},\boldsymbol{X}_{i,j}) = \arg\max_{\boldsymbol{\beta}\in\Xi} \sum_{j=1}^{m}\sum_{i=1}^{n} \left[Y_{i,j}\boldsymbol{X}_{i,j}'\boldsymbol{\beta}_i - H(\boldsymbol{X}_{i,j}'\boldsymbol{\beta}_i)\right] \tag{6}$$
$$\text{where } \Xi := \{\boldsymbol{\beta} : \beta_{i,l} = \beta_{i',l} \text{ if (i) } l \in D_s \text{ or (ii) } l \in D_c \text{ and } \varsigma(i,l) = \varsigma(i',l)\}.$$

Thus, $\hat{b}^a_{i,l}$ is the estimated coefficient of feature l for item i under the aggregate approach. As a counterpart of Proposition 1, the following result establishes the consistency and normality of the aggregate estimator $\hat{\boldsymbol{b}}^a$.

PROPOSITION 2. *The aggregate estimator $\hat{\boldsymbol{b}}^a$ satisfies the following properties:*

(a) CONSISTENCY. *As $m \uparrow +\infty$, (i) if $l \in \mathcal{D}_s$, then $\hat{b}^a_{i,l}(m) \xrightarrow{p} \beta^s_l$; if $l \in \mathcal{D}_n$, then $\hat{b}^a_{i,l}(m) \xrightarrow{p} \beta^n_{i,l}$; and (iii) if $l \in \mathcal{D}_c$, then $\hat{b}^a_{i,l}(m) \xrightarrow{p} \beta^c_{\varsigma(i,l),l}$.*

(b) NORMALITY. *For each item i and each feature $l \in \mathcal{D}$, there exist a threshold $\tilde{\mathfrak{m}}_{i,l}$ on the sample size and a constant $\tilde{\psi}_{i,l} > 0$ such that if $m \geq \tilde{\mathfrak{m}}_{i,l}$, we have, for any $\epsilon > 0$,*

$$\mathbb{P}(|\hat{b}^a_{i,l}(m) - \beta_{i,l}| \leq \epsilon) \geq 1 - 3\exp(-\tilde{\psi}_{i,l}\epsilon^2 m).$$

Furthermore, if $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$, then $\sqrt{m}(\hat{\boldsymbol{b}}^a(m) - \boldsymbol{\beta})$ converges in distribution to a zero-mean degenerate multivariate normal distribution as $m \uparrow +\infty$.

Although the aggregate model requires knowing the data aggregation level of each feature and the cluster structure of the items, Proposition 1 facilitates us to infer such critical information with high accuracy from the statistical properties of the decentralized approach. We are now ready to introduce the *Data Aggregation with Clustering* (DAC) algorithm (described in Algorithm 1), which consistently estimates the coefficient of each feature for each item, as well as correctly identifies the aggregation levels and the underlying cluster structure of the items. We denote the CDF of a standard normal distribution by $\Phi(\cdot)$.

Algorithm 1 Data Aggregation with Clustering DAC_α

Initialize: α =the significance level for hypothesis testing; $\hat{\mathcal{D}}_n = \hat{\mathcal{D}}_s = \hat{\mathcal{D}}_c = \emptyset$.

1: (Decentralized Estimation) Estimate $\hat{b}_{i,l}$ and its standard error $\widehat{SE}_{i,l} = \sqrt{\frac{1}{m}(\hat{\mathcal{I}}_i(\hat{\boldsymbol{b}}_i)^{-1})_{l,l}}$ for each item i and feature l, where $\hat{\mathcal{I}}_i(\hat{\boldsymbol{b}}_i)$ is the estimated Fisher information matrix for item i.

For each feature $l \in D$**,**

2: (Hypothesis Testing) Fix an item 1. For all other items $i \neq 1$, compute the p-value under the null hypothesis $H^l_{1,i}$ that $b_{1,l} = b_{i,l}$, which we denote as

$$p_{1,i} = 2\Big[1 - \Phi\Big(\frac{|\hat{b}_{1,l} - \hat{b}_{i,l}|}{\sqrt{\widehat{SE}^2_{1,l} + \widehat{SE}^2_{i,l}}}\Big)\Big].$$

Reject $H^l_{1,i}$ if and only if $p_{1,i} < \alpha$.

3: If $H^l_{1,i}$ is not rejected for all items $i \neq 1$, then assign feature l to $\hat{\mathcal{D}}_s$.

4: If $H^l_{1,i}$ is rejected for some items $i \neq 1$ and not for others, then assign feature l to $\hat{\mathcal{D}}_c$.

5: If $H^l_{1,i}$ is rejected for all items $i \neq 1$, then assign feature l to $\hat{\mathcal{D}}_n$.

6: (Clustering) If $l \in \hat{\mathcal{D}}_c$, run a one-dimensional k-means algorithm on $\{\hat{b}_{1,l}, \hat{b}_{2,l}, ..., \hat{b}_{n,l}\}$ with $k = k_l$ and obtain the estimated cluster structure $\{\hat{\mathcal{C}}_{1,l}, \hat{\mathcal{C}}_{2,l}, \ldots, \hat{\mathcal{C}}_{k_l,l}\}$.

End For.

7: (Aggregate Estimation) Obtain the aggregate estimator $\hat{\boldsymbol{b}}^a$ based on the aggregation levels $(\hat{\mathcal{D}}_n, \hat{\mathcal{D}}_s, \hat{\mathcal{D}}_c)$ and the cluster structures $\{\hat{\mathcal{C}}_{1,l}, \hat{\mathcal{C}}_{2,l}, \ldots, \hat{\mathcal{C}}_{k_l,l}\}$ for each $l \in \hat{\mathcal{D}}_c$.

Output: (a) Aggregation levels: $(\hat{\mathcal{D}}_n, \hat{\mathcal{D}}_s, \hat{\mathcal{D}}_c)$, (b) Cluster structures: $\{\hat{\mathcal{C}}_{1,l}, \hat{\mathcal{C}}_{2,l}, \ldots, \hat{\mathcal{C}}_{k_l,l}\}$ for each $l \in \hat{\mathcal{D}}_c$, and (c) Feature coefficients: $\hat{\boldsymbol{b}}^a$.

Throughout this paper, we use DAC_α to denote the Data Aggregation with Clustering algorithm initialized with a significance level α and $\hat{\boldsymbol{\beta}}^\alpha$ the estimated feature coefficient matrix of the DAC_α algorithm. We also call $\hat{\boldsymbol{\beta}}^\alpha$ the DAC_α estimator. By leveraging the consistency and normality of the decentralized estimate (see Proposition 1), we can perform hypothesis testing (i.e., the Wald test) to identify the correct data aggregation levels and cluster structure.[1] The main idea is as follows: if one cannot reject the null hypothesis that the estimated coefficients in the decentralized model $\hat{b}_{i,l}$ and $\hat{b}_{i',l}$ are the same, then it is very likely that either items i and i' belong to the same cluster or that feature l is an aggregate-level feature. An interesting characteristic of our method is that it uses the estimated coefficients as inputs to identify the cluster structure of the items with respect to each cluster-level feature (see Step 6), as opposed to directly using features as in traditional clustering algorithms. We note that identifying the cluster structure in Step 6 of Algorithm 1 is very efficient since this step runs a single-dimensional k-means. For each item i and feature $l \in \mathcal{D}$, we denote the estimated cluster that the item belongs to as $\hat{\mathcal{C}}(i,l)$. We also highlight that the last step of the DAC algorithm to fit an aggregate model can be regularized using a Lasso, Ridge, or elastic net penalty

[1] Under Assumption 1(b), each cluster has at least two items. Thus, Step 4 of Algorithm 1 (with a time complexity of $O(nd)$) is sufficient to determine the SKU-level features. If we relax this assumption (i.e., some cluster may have only one feature), the DAC algorithm can easily be adapted to run $O(n^2 d)$ pairwise hypothesis tests instead of $O(nd)$.

to avoid unnecessary features and mitigate overfitting. This would be especially useful with high correlations between features, which is common in retail settings. Finally, we remark that the DAC algorithm is in a similar spirit as the two-stage heuristic proposed by Park et al. (2017) to address a special case of our problem, which has only cluster-level features and assumes identical cluster structures for all features. The two-stage heuristic fits the decentralized model in the first stage and then runs a multi-dimensional clustering algorithm for all the decentralized coefficients in the second stage. Our main contribution relative to this two-stage heuristic is multi-faceted: (a) we adopt hypothesis testing to account for different data aggregation levels across features; (b) we account for different cluster structures across features; (c) we provide rigorous theoretical justifications for the validity of our approach and the edge of our method over the decentralized benchmark.

We next show that, with an arbitrarily high probability, the DAC_α algorithm can consistently identify the aggregation level of each feature, as well as the underlying cluster structure of the items with respect to each cluster-level feature. Under the DAC_α algorithm and sample size m, we define $\mathcal{E}(m,\alpha)$ as the event where the aggregation level of each feature and the cluster structure of each item for each cluster-level feature is correctly identified, namely, (i) $(\hat{\mathcal{D}}_n,\hat{\mathcal{D}}_s,\hat{\mathcal{D}}_c)=(\mathcal{D}_n,\mathcal{D}_s,\mathcal{D}_c)$ and (ii) $(\hat{\mathcal{C}}_{1,l},\hat{\mathcal{C}}_{2,l},\ldots,\hat{\mathcal{C}}_{k_l,l})$ is a permutation of $(\mathcal{C}_{1,l},\mathcal{C}_{2,l},\ldots,\mathcal{C}_{k_l,l})$ for each $l\in\mathcal{D}_c$. Consistent with the decentralized and aggregate approaches, the DAC_α algorithm generates a consistent and normal estimator $\hat{\boldsymbol{\beta}}^\alpha$.

PROPOSITION 3. *The* DAC_α *algorithm satisfies the following properties:*

(a) CONSISTENCY. *There exists a positive probability* $p(\alpha)$ *strictly decreasing in* α *with* $\lim_{\alpha\downarrow 0}p(\alpha)=0$*, such that*

$$\lim_{m\uparrow\infty}\mathbb{P}\Big[\mathcal{E}(m,\alpha)\Big]\geq 1-p(\alpha). \tag{7}$$

Furthermore, $\hat{\boldsymbol{\beta}}^\alpha$ *is consistent, that is, as* $m\uparrow+\infty$*, (i) if* $l\in\mathcal{D}_s$*, then* $\hat{\beta}^\alpha_{i,l}(m)\xrightarrow{p}\beta^s_l$*; if* $l\in\mathcal{D}_n$*, then* $\hat{\beta}^\alpha_{i,l}(m)\xrightarrow{p}\beta^n_{i,l}$*; and (iii) if* $l\in\mathcal{D}_c$*, then* $\hat{\beta}^\alpha_{i,l}(m)\xrightarrow{p}\beta^c_{\varsigma(i,l),l}$*.*

(b) NORMALITY. *For each item* i *and each feature* $l\in\mathcal{D}$*, there exist a threshold* $\mathfrak{m}^\alpha_{i,l}$ *on the sample size and two constants* $\psi^\alpha_{i,l}>0$ *and* $\eta^\alpha_{i,l}>0$*, such that if* $m\geq\mathfrak{m}^\alpha_{i,l}$*, we have, for any* $\epsilon>0$*,*

$$\mathbb{P}(|\hat{\beta}^\alpha_{i,l}(m)-\beta_{i,l}|\leq\epsilon)\geq 1-\eta^\alpha_{i,l}\exp(-\psi^\alpha_{i,l}\epsilon^2 m). \tag{8}$$

Misspecifying the feature aggregation levels for the DAC_α algorithm may stem from two types of errors in hypothesis testing (Step 2 in Algorithm 1): (i) Type-I errors (i.e., rejecting the otherwise true null hypothesis) under which the algorithm falsely identifies two identical coefficients as different from each other, and (ii) Type-II errors (i.e., not rejecting the otherwise false null hypothesis) under which the algorithm falsely identifies two different coefficients to be the same. By the finite-sample and asymptotic normality of the decentralized estimator (i.e., Proposition 1(b)), the Type-II errors

of DAC_α decay exponentially with the sample size m, which shrink to 0 as m approaches infinity. The Type-I errors of the DAC_α algorithm are induced by the errors in each hypothesis test, which are controlled by the significance level α. Although the Type-I error probability for each $H^0_{1,i}$ is upper bounded by α, due to the notorious multiple hypothesis testing issue (see, e.g., Shaffer 1995), the total Type-I error probability of the DAC_α algorithm is in general higher than α. Indeed, we leverage the asymptotic normality of the decentralized estimator to evaluate this probability as $p(\alpha)$, which can be arbitrarily small with a proper choice of the significance level α (see the proof of Proposition 3 for all the details). Equivalently, we can also follow adjust p-values via Bonferroni correction from the simultaneous inference literature (see, e.g., Shaffer 1995) to implement the $\mathsf{DAC}_{\alpha(p)}$ algorithm. In this case, the total Type-I error probability is upper bounded by p, where $\alpha(p)$ is the (strictly decreasing) inverse of $p(\alpha)$. For the practical implementation of the DAC_α algorithm using data, as we demonstrate in Sections 5 and 6, the significance level α for a single hypothesis test is a hyper-parameter to be fine-tuned via cross-validation. Finally, we remark that, for our demand prediction problem with heterogeneous data aggregation levels, Type-I errors are somehow acceptable in the sense that they will only cause model imprecision (i.e., the algorithm does not identify identical coefficients), but not misspecification, so that the estimation remains consistent and normal even under these errors (Proposition 3(a) and (b)). Instead, Type-I errors will only affect the efficiency of the DAC estimator, giving rise to estimates with a higher variance. We will elaborate on this point in Section 4 below.

Several remarks are in order regarding the DAC_α algorithm and its consistency and normality. First, Proposition 3 shows the effectiveness of Algorithm 1 under the regime where the sample size m is sufficiently large. For the case where the data sample size m is small (e.g., a new item with limited data), the estimation variance will be high. In this case, the DAC algorithm, based on hypothesis testing, may misspecify the aggregation level of each feature, giving rise to Type-I errors, Type-II errors, and eventually prediction errors. To address this model misspecification issue, we adopt a cross-validation procedure to fine-tune several hyper-parameters, including the significance level α for hypothesis testing and the threshold for classifying each feature into clustering and department levels (see Sections 5 and 6 for the implementation details). Using both simulated and real data, in Sections 5 and 6, we show that our proposed DAC algorithm outperforms a multitude of well-established benchmarks used in the literature and in practice. We also examine the case with new items that have limited data availability. Our theoretical results (Proposition 6) and our computational results (Figure 3) show that our DAC algorithm can efficiently pool and leverage the data, while substantially improving the prediction accuracy for *all* items regardless of their data availability. More importantly, the accuracy improvement is more substantial for items with limited data.

Second, we assume in our base model that all the observations are independent across time periods and items. Such an independence assumption can also be relaxed through the vector auto-regression (VAR) model for the case with correlations across time periods and through the generalized least squares (GLS) model for the case with correlations across items. We conclude this section by discussing how the DAC algorithm can be generalized to these two cases.

Correlation across time periods. To account for correlations across time periods, we assume that the observations $\{1,2,\ldots,m\}$ are ranked in chronological order. More precisely, the data collected in period j are indexed as observation j. We consider the following VAR model specification:

$$Y_{i,j} = \rho_i Y_{i,j-1} + \sum_{l\in\mathcal{D}_s} X^l_{i,j}\beta^s_l + \sum_{l\in\mathcal{D}_n} X^l_{i,j}\beta^n_{i,l} + \sum_{l\in\mathcal{D}_c} X^l_{i,j}\beta^c_{\varsigma(i,l)} + \epsilon_{i,j},\ i=1,\ldots,n \text{ and } j=2,\ldots,m,$$

where $\rho_i Y_{i,j-1}$ is the auto-regression term, β^s_l, $\beta^n_{i,l}$, and $\beta^c_{\varsigma(i,l)}$ are defined in the same fashion as in the base model, and $\epsilon_{i,j}$ are the independent unobservable zero-mean sub-Gaussian error terms uncorrelated with the features $\boldsymbol{X}$ (i.e., $\mathbb{E}[\epsilon_{i,j}|\boldsymbol{X}]=0$). Applying the standard VAR estimation approach (see, e.g., Chapter 19 of Greene 2003), we can follow the same procedure as Algorithm 1 to estimate the decentralized model, conduct hypothesis tests, cluster the items, and, finally, estimate the aggregate model. All the results presented in this paper remain valid under this VAR setting.

Correlation across items. To account for correlations across items, we consider the following GLS model specification:

$$Y_{i,j} = \sum_{l\in\mathcal{D}_s} X^l_{i,j}\beta^s_l + \sum_{l\in\mathcal{D}_n} X^l_{i,j}\beta^n_{i,l} + \sum_{l\in\mathcal{D}_c} X^l_{i,j}\beta^c_{\varsigma(i,l)} + \epsilon_{i,j},\ i=1,\ldots,n \text{ and } j=1,\ldots,m,$$

where β^s_l, $\beta^n_{i,l}$, and $\beta^c_{\varsigma(i,l)}$ are defined in the same fashion as in the base model, and $\epsilon_{i,j}$ are the unobservable error terms with $\mathbb{E}[\epsilon_{i,j}|\boldsymbol{X}]=0$, but $\mathbb{E}[\epsilon_{i,j}\epsilon_{i',j}|\boldsymbol{X}]\neq 0$ for $i\neq i'$. This model extension allows us to capture *complementary* and *substitute* relationships between different SKUs within a department, such as the *cannibalization effect* between dark roast and medium roast coffee and the *halo effect* between shampoo and conditioner. For this model, one can slightly modify the DAC algorithm to correctly estimate the aggregation levels, cluster structures, and feature coefficients. More specifically, we follow the same procedure as in Steps 1–6 of Algorithm 1 to estimate the decentralized model, conduct hypothesis tests, and cluster the items. However, the final step (Step 7) that estimates the aggregate model should be adjusted to apply the GLS (instead of OLS) estimation method (see, e.g., Chapter 10 of Greene 2003). All the results presented in this paper remain valid under this GLS setting. If there are correlations across both time periods and items, we could combine the VAR and GLS frameworks and modify the DAC algorithm accordingly to yield consistent estimates of the model coefficients, data aggregation levels, and cluster structure.

4. The Value of Pooling Data Through DAC

The remainder of this paper is devoted to characterizing the benefits of performing the pairwise tests and the clustering algorithm benchmarked against existing approaches in the literature. Specifically, we present three sets of analyses from different perspectives: (a) analytical comparisons between the DAC algorithm and the decentralized method, which highlight the value of data aggregation through efficient clustering; (b) simulation studies of the DAC algorithm versus several benchmarks, which show that the DAC algorithm can successfully identify and leverage data aggregation and the cluster structures; and (c) implementation of the DAC algorithm using real retail data, which showcases the practical value of our proposed method in improving demand prediction accuracy.

In this section, we examine the value of data aggregation through efficient clustering from a theoretical perspective by showing several benefits of the aggregated model relative to the decentralized model. To convey the DAC algorithm's benefits, we first observe that if the true data-generating process has aggregate and/or cluster level features, the decentralized model assumes an overly complex model and, hence, will be prone to overfitting. To formalize this intuition, we leverage the asymptotic normality property of the MLE established by Propositions 1, 2, and 3 to derive the following result on the estimation errors of the decentralized approach and the DAC algorithm.

PROPOSITION 4. *The following statements hold:*

(a) For the decentralized estimator $\hat{\boldsymbol{b}}$*, there exists a constant* $\kappa_{i,l} = (\mathcal{I}_i(\boldsymbol{\beta}_i)^{-1})_{l,l} > 0$ *for each* (i,l) *such that*

$$\lim_{m\uparrow+\infty} m \cdot \mathbb{E}(\hat{b}_{i,l}(m) - \beta_{i,l})^2 = \kappa_{i,l}, \qquad i = 1,2,\ldots,n,\ l = 1,2,\ldots,d. \tag{9}$$

(b) For the DAC_α *estimator* $\hat{\boldsymbol{\beta}}^\alpha$*, under the same set of constants* $\{\kappa_{i,l} : 1 \le i \le n, 1 \le l \le d\}$*, we have*

$$\lim_{m\uparrow+\infty} m \cdot \mathbb{E}(\hat{\beta}^\alpha_{i,l}(m) - \beta_{i,l})^2 \le p(\alpha)\kappa_{i,l} + (1-p(\alpha))\Big(\frac{1}{n_{i,l}}\Big)^2\Big(\sum_{i'\in\mathcal{C}(i,l)} \kappa_{i',l}\Big),\ i = 1,2,\ldots,n,\ l = 1,2,\ldots,d, \tag{10}$$

where $p(\alpha)$ *is the upper bound on the total Type-I error of* DAC_α *characterized in Proposition 3 and the inequality is strict if* $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$*.*

(c) If $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$ *and* $\kappa_{i,l} > \Big(\frac{1}{n_{i,l}}\Big)^2\Big(\sum_{i'\in\mathcal{C}(i,l)} \kappa_{i',l}\Big)$*, there exists a threshold* $\bar{\mathfrak{m}}$ *such that if* $m \ge \bar{\mathfrak{m}}$*, we have*

$$\mathbb{E}(\hat{\beta}^\alpha_{i,l}(m) - \beta_{i,l})^2 < \mathbb{E}(\hat{b}_{i,l}(m) - \beta_{i,l})^2. \tag{11}$$

By the consistency of the decentralized and DAC estimators, when the number of observations m becomes large, the expected squared error of the estimated coefficient for each item and feature will shrink to zero. The positive constant $\kappa_{i,l} = (\mathcal{I}_i(\boldsymbol{\beta}_i)^{-1})_{l,l}$ is the asymptotic variance of the decentralized estimator $\hat{b}_{i,l}$. What makes the DAC estimator more powerful is its ability to pool the data from

different items, and thus further reduce the variance of the estimates, as shown by the comparison result in Proposition 4(c). We note that $\frac{1}{n_{i,l}} \sum_{i' \in \mathcal{C}(i,l)} \kappa_{i',l}$ is the average variance of the estimated coefficient of feature l for items in $\mathcal{C}(i,l)$ when using the decentralized estimator. Since $n_{i,l} \geq 2$ for $l \in D_s \cup D_c$, the condition $\kappa_{i,l} > \left(\frac{1}{n_{i,l}}\right)^2 \left(\sum_{i' \in \mathcal{C}(i,l)} \kappa_{i',l} \right)$ can easily be satisfied. Furthermore, the Type-I error probability is upper bounded by $p(\alpha) \approx 0$, so that the DAC algorithm yields a smaller asymptotic error relative to the decentralized estimator for the coefficients of features at the department or cluster levels. In this case, for the same training sample, the DAC algorithm will use at least twice as many observations as the decentralized estimator to estimate the coefficient of such features. Hence, the estimation error will shrink to zero faster, especially when $n_{i,l}$ is large. In practice, a typical retail department consists of a large number of items ($n > 100$), so the DAC algorithm will be much more efficient relative to the decentralized estimator for cluster- and aggregate-level features. Moreover, for these features, the value of the DAC algorithm in reducing the estimation variance of their coefficients will be stronger when the number of items n increases, because it allows to pool more data to improve the estimation efficiency of these features.

One can see from Eq. (10) that the estimation error of the DAC algorithm can be decomposed into two parts. The first part, $p(\alpha)\kappa_{i,l}$, bounds the error for the case when a Type-I error occurs so that the algorithm fails to identify the pooled feature coefficients. In this case, the estimation error is upper bounded by the error of the decentralized approach, $\kappa_{i,l}$, multiplied by the maximum chance that this case occurs $p(\alpha)$. The second part, $(1-p(\alpha))\left(\frac{1}{n_{i,l}}\right)^2 \left(\sum_{i' \in \mathcal{C}(i,l)} \kappa_{i',l} \right)$ bounds the error for the case where the DAC algorithm correctly identifies the data aggregation levels and the cluster structures. In this case, the estimation error is upper bounded by the error of the aggregate estimator, $\left(\frac{1}{n_{i,l}}\right)^2 \left(\sum_{i' \in \mathcal{C}(i,l)} \kappa_{i',l} \right)$, multiplied by the chance that the data aggregation levels and the cluster structure are correctly specified, $1-p(\alpha)$. As long as there are some aggregate level or cluster level features (i.e., $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$), then the DAC algorithm could at least partially identify some (but not all) of the pooled coefficients so that the expected error will be strictly smaller relative to the decentralized model even if a Type-I error occurs. Hence, the inequality in Eq. (10) is strict when $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$. Finally, we remark that the expected squared error driven Type-II errors of the DAC algorithm is of a lower magnitude relative to the advantage of the aggregate estimator over the decentralized and, thus, can be ignored without compromising the overall error bound of the algorithm.

Proposition 4 demonstrates the significant efficiency improvement (i.e., reducing the expected estimation error) of the DAC algorithm relative to the decentralized estimator under the presence of aggregate- and cluster-level features. We next analyze how the DAC algorithm affects the mean squared error (MSE) of the predicted outcome under a fixed design (i.e., viewing the feature matrix $\boldsymbol{X}$ as *deterministic*; see, e.g., Chapter 2 of Rigollet 2015) for the linear regression model (i.e., $G(u) = u$

and $\epsilon_{i,j}$ follows an *i.i.d.* normal distribution with mean 0 and standard deviation σ). For any estimator $\hat{\boldsymbol{\beta}}$, we define its MSE as

$$\widehat{\mathcal{MSE}}(\hat{\boldsymbol{\beta}}) := \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}(Y_{i,j} - \boldsymbol{X}_{i,j}\hat{\boldsymbol{\beta}}_i)^2}{nm}.$$

PROPOSITION 5. *Under a linear regression setting, the following statements hold:*

(a) The MSE of the decentralized estimator $\hat{\boldsymbol{b}}$ satisfies

$$\mathbb{E}\left[\widehat{\mathcal{MSE}}(\hat{\boldsymbol{b}})\right] \le \frac{4\sigma^2 d}{m}, \tag{12}$$

where the expectation is taken with respect to the error terms $\epsilon_{i,j}$.

(b) Assume that $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$. The MSE of the DAC$_\alpha$ *estimator $\hat{\boldsymbol{\beta}}^\alpha$ satisfies*

$$\mathbb{E}\left[\widehat{\mathcal{MSE}}(\hat{\boldsymbol{\beta}}^\alpha)\right] \le \frac{4\sigma^2 d_x(\alpha)}{nm} + o(m^{-1}) < \frac{4\sigma^2 d}{m} \tag{13}$$

when m is sufficiently large. Here, the expectation is taken with respect to both the randomness of the error terms $\epsilon_{i,j}$ and the DAC *algorithm, $d_x(\alpha) < nd$ is the expected number of coefficients to estimate in Step 7 of Algorithm 1, which is decreasing in α and such that $\lim_{\alpha \downarrow 0} d_x(\alpha) = d_x$, and $o(m^{-1})$ is the standard "Little-o Notation".*

Comparing parts (a) and (b) of Proposition 5 reveals the insight that the expected MSE of the DAC estimator is substantially lower relative to the decentralized benchmark. Such an improvement is driven by the fact that our proposed DAC algorithm leverages hypothesis testing to pool data, hence significantly reducing the number of model coefficients to estimate (from nd to $d_x(\alpha)$), which corresponds to 20%–70%, as shown by the implementation of our algorithm using real data (see Table 7). This illustrates the power of aggregating data and reducing the model dimensionality in order to ultimately improve prediction accuracy.

We next study an important generalization of our base model where each item i has a different number of observations m_i. This setting fits well the scenario where some items have been offered for a long time (and, hence, have abundant data), whereas other items are new to the market (and, hence, have limited data). More specifically, we assume that, in the training set, item i has $m_i = m\tau_i$ observations for each i. Namely, the larger τ_i, the more data is available for item i. We define $\tau := \sum_{i=1}^{n} \tau_i$ and $\tau(i,l) := \sum_{i' \in \mathcal{C}(i,l)} \tau_{i'}$. Hence, $\tau(i,l) = \tau$ if $l \in \mathcal{D}_s$ and $\tau(i,l) = \tau_i$ if $l \in \mathcal{D}_n$. Then, the total number of data observations with the same coefficient for feature l as $\beta_{i,l}$ is $m(i,l) := m \cdot \tau(i,l)$. To highlight the main intuition without getting trapped in technical details, we focus on the linear regression setting with independent features. Namely, we assume that for each item i and for all data observations j, $X_{i,j}^l$ are *i.i.d.* with mean 0 and variance 1. The entire training dataset is denoted as $\mathfrak{D}_{\mathsf{tr}} := \{(Y_{i,j}\boldsymbol{X}_{i,j}) : 1 \le i \le n, 1 \le j \le m_i\}$, whereas the training dataset of item i is denoted as $\mathfrak{D}_{\mathsf{tr}}(i) := \{(Y_{i,j}\boldsymbol{X}_{i,j}) : 1 \le i \le n, 1 \le j \le m_i\}$.

We are interested in the *generalization error* (GE) of each item under different demand prediction methods in the random design setting, which measures the out-of-sample expected squared error assuming that the training and testing data are independently drawn from the same data generating process (see, e.g., Chapter 2.9 of Hastie et al. 2019). For an estimation algorithm $\pi \in \{\mathsf{Dec}, \mathsf{DAC}_\alpha\}$ trained on $\mathfrak{D}_{\mathsf{tr}}$, where Dec refers to the decentralized approach, we define the estimator generated by π as $\hat{\boldsymbol{\beta}}(\pi, \mathfrak{D}_{\mathsf{tr}}) = (\hat{\boldsymbol{\beta}}_1(\pi, \mathfrak{D}_{\mathsf{tr}}), \hat{\boldsymbol{\beta}}_2(\pi, \mathfrak{D}_{\mathsf{tr}}), ..., \hat{\boldsymbol{\beta}}_n(\pi, \mathfrak{D}_{\mathsf{tr}}))$, where $\hat{\boldsymbol{\beta}}_i(\pi, \mathfrak{D}_{\mathsf{tr}})$ is the coefficient vector for item i. The GE of π for each item i is defined as follows:

$$\mathcal{GE}_i(\pi) := \mathbb{E}\left[Y_{i,m_i+1} - \boldsymbol{X}_{i,m_i+1}\hat{\boldsymbol{\beta}}_i(\pi, \mathfrak{D}_{\mathsf{tr}})\right]^2, \qquad i = 1, 2, ..., n,$$

where the testing data $(Y_{i,m_i+1}, \boldsymbol{X}_{i,m_i+1})$ is independently drawn from the same distribution as each training observation $(Y_{i,j}, \boldsymbol{X}_{i,j}) \in \mathfrak{D}_{\mathsf{tr}}(i)$ and the expectation is taken with respect to the randomness of both the training and testing data.

The following proposition demonstrates the effectiveness of the DAC algorithm (in the asymptotic regime) for the case where different items have different data volumes.

PROPOSITION 6. *Under a linear regression setting, the following statements hold if the estimators are trained on* $\mathfrak{D}_{\mathsf{tr}}$*:*

(a) The GE of the Dec *estimator satisfies*

$$\lim_{m\uparrow+\infty} m \cdot \left(\mathcal{GE}_i(\mathsf{Dec}) - \sigma^2\right) = \frac{d \cdot \sigma^2}{\tau_i}, \qquad i = 1, 2, ..., n. \tag{14}$$

(b) The GE of the DAC_α *estimator satisfies*

$$\lim_{m\uparrow+\infty} m \cdot \left(\mathcal{GE}_i(\mathsf{DAC}_\alpha) - \sigma^2\right) \le p(\alpha) \cdot \frac{d \cdot \sigma^2}{\tau_i} + (1 - p(\alpha)) \cdot \left(\sum_{l=1}^{d} \frac{1}{\tau(i,l)}\right) \cdot \sigma^2, \qquad i = 1, 2, ..., n, \tag{15}$$

where $p(\alpha)$ *is the upper bound of the Type-I error of* DAC_α *characterized in Proposition 3 and the inequality is strict if* $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$*.*

(c) If $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$*, we have, for each item* $i = 1, 2, ..., n$*, and sufficiently large* m*,*

$$m \cdot \left(\mathcal{GE}_i(\mathsf{Dec}) - \mathcal{GE}_i(\mathsf{DAC}_\alpha)\right) > (1 - p(\alpha)) \cdot \left(d_s\left(\frac{1}{\tau_i} - \frac{1}{\tau}\right) + \sum_{l \in \mathcal{D}_c}\left(\frac{1}{\tau_i} - \frac{1}{\tau(i,l)}\right)\right) \cdot \sigma^2 > 0. \tag{16}$$

Furthermore, $g_i(\boldsymbol{\tau}) := d_s\left(\frac{1}{\tau_i} - \frac{1}{\tau}\right) + \sum_{l \in \mathcal{D}_c}\left(\frac{1}{\tau_i} - \frac{1}{\tau(i,l)}\right)$ *is convexly decreasing in* τ_i *and concavely increasing in* $\tau_{i'}$ *for each* i *and* $i' \neq i$*.*

Proposition 6 shows that when the sample size is sufficiently large, the DAC algorithm yields a lower generalization error relative to the decentralized approach for each item, regardless of the item's data availability. Furthermore, pooling data via our proposed DAC algorithm reduces the generalization error more substantially for items with a lower amount of data (i.e., smaller τ_i). As

expected, the prediction accuracy improvement of the DAC algorithm is strengthened when the number of features at the aggregate or cluster level, d_s or d_c, is higher. We further show the robustness of this result via extensive computational experiments in Section 5.2. The monotonicity result of $g_i(\boldsymbol{\tau})$ in Proposition 6(c) also indicates that if the sample size of one item increases, other items can successfully leverage this fact but with diminishing marginal returns, whereas the item itself could extract less additional information from other items' data. The proof of Proposition 6 relies on the bias-variance decomposition of the linear regression model. We also highlight that, similar to the estimation error characterized in Proposition 4, the generalization error of the DAC algorithm can be decomposed into two parts, where the first part, $p(\alpha)\cdot\frac{d\cdot\sigma^2}{\tau_i}$, measures the GE when Type-I errors occur, and the second part, $(1-p(\alpha))\cdot\left(\sum_{l=1}^{d}\frac{1}{\tau(i,l)}\right)\cdot\sigma^2$, measures the GE when the DAC algorithm correctly identifies aggregation levels and cluster structures.

To conclude this section, we note that we have focused our analysis on the case where the sample size m is sufficiently large. In practice, the sample size is often limited. Ultimately, one may question whether the value of pooling data via our proposed DAC algorithm remains significant in the small-sample regime. Our simulation and real data studies (Sections 5 and 6) clearly convey that the DAC algorithm efficiently identifies and leverages data aggregation and, hence, substantially improves the out-of-sample prediction accuracy relative to several common benchmarks.

5. Simulated Experiments

In this section, we conduct computational experiments using simulated data. We focus on the DAC algorithm's predictive power and illustrate the improvement in prediction accuracy relative to several benchmarks in the non-asymptotic regime (i.e., when the data sample size m is moderate or small). We consider two GLM settings: linear regression (i.e., $G(u)=u$; see Section 5.1) and logistic regression (i.e., $G(u)=1/(1+\exp(-u))$; see Section 5.3). The model's performance is evaluated using the out-of-sample R^2 for linear regression and the area under the ROC curve (AUC) for logistic regression. We also undertake a comprehensive sensitivity analysis to investigate how the different parameters affect the model's performance. All the results presented in this section can be reproduced by using our available code and implementation details.[2]

5.1. Linear Regression

We assume that the data is generated from the following linear model:

$$Y_{i,j}=\sum_{l\in\mathcal{D}_s}X^l_{i,j}\beta^s_l+\sum_{l\in\mathcal{D}_n}X^l_{i,j}\beta^n_{i,l}+\sum_{l\in\mathcal{D}_c}X^l_{i,j}\beta^c_{\varsigma(i,l),l}+\epsilon_{i,j},\qquad i=1,2,\ldots,n \text{ and } j=1,2,\ldots,m,$$

[2] https://github.com/DACPublicator/DAC_Publication

Table 1 Parameters used in Section 5.1.

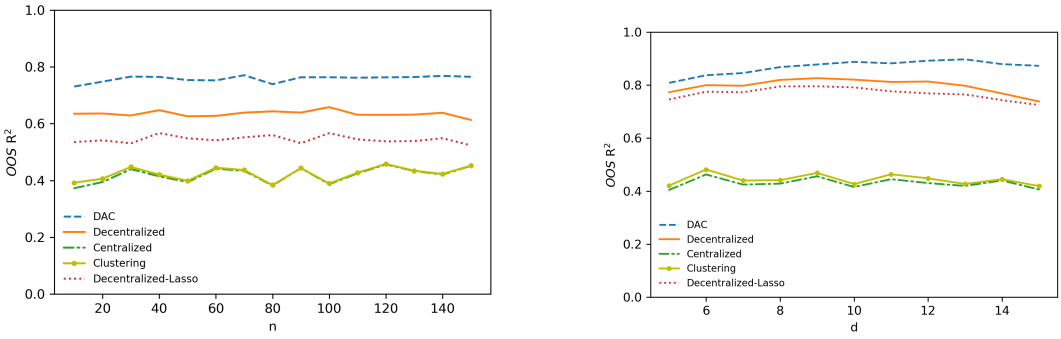
Parameter	Range of values
Number of items (n)	$[10, 150]$
Number of features (d)	$[5, 15]$
Number of observations (m)	$[10, 50]$
Variance of the noise (σ^2)	$[0.25, 2]$
Department-level probability (p)	$[1/6, 2/3]$ or $[1/6, 1/3]$
Cluster-level probability (q)	$[1/6, 2/3]$ or $[1/6, 1/3]$

where $\epsilon_{i,j} \sim \mathcal{N}(0, \sigma^2)$ are i.i.d. random variables. Each feature, $X^l_{i,j}$, is generated randomly from a uniform $[0, 1]$ distribution, and each coefficient β is obtained from a uniform $[-5, 5]$ distribution. We fix the number of clusters $k_l = 2$ for all feature $l \in \mathcal{D}$ and vary the parameters $\{n, d, m, \sigma^2, p, q\}$ one at a time (we observed similar results for alternative values of k_l's). The definition and range of values for these parameters are reported in Table 1. The parameters p and q represent the probability that a given feature is modeled at the aggregate and cluster levels, respectively (different features are drawn independently). Thus, the probability that a feature is at the SKU level is $(1 - p - q)$.

It is important to note that the DAC algorithm's implementation admits three hyper-parameters (i.e., θ, R_U, and R_L) in addition to the numbers of clusters k_l's. These three parameters represent the strictness of our algorithm in determining whether a feature should be aggregated. Specifically, $\theta = \alpha$ is the p-value cutoff for statistical significance and is usually set at 0.05 or 0.01. The parameters R_U and R_L represent thresholds for the ratio of non-rejected hypotheses. For example, suppose that the percentage of non-rejected hypotheses for feature j is $R_j = 0.3$ (i.e., 30% of the items have statistically close estimated coefficients). Then, we label feature j as a department-level feature if $R_j > R_U$ and as a SKU-level feature if $R_j < R_L$. For any intermediate value $R_j \in [R_L, R_U]$, we will label feature j as a cluster-level feature. A good way to set the parameters R_L and R_U is by performing a cross-validation procedure (for more details, see Section 6). The parameters θ, R_U, and R_L provide flexibility in the tolerance level of the algorithm. In this section, we fine-tune these parameters by testing a grid of values. When implementing our algorithm with real data (Section 6), we will carefully set their values using a rigorous cross-validation procedure.

To test the performance of the DAC algorithm, we consider the following four benchmarks: decentralized, decentralized-Lasso, centralized, and clustering. For each problem instance (i.e., a specific combination of $\{n, d, m, \sigma, p, q\}$), we generate 100 independent trials (i.e., datasets) and use 67% as training and 33% as testing for each trial. We then report the average out-of-sample R^2 across all items and observations. Below is a description of the methods we consider:

1. DAC: We implement our algorithm with $\theta = 0.05$, $R_U = 0.9$, and $R_L = 0.6$.
2. Decentralized: We estimate a simple OLS model for each item separately (i.e., n models).
3. Decentralized-Lasso: Same as the decentralized method, but we add an ℓ_1 regularization term to each OLS model (we obtained similar results when using a regularization based on Ridge regression or Elasticnet).



(a) Varying the number of items

(b) Varying the number of features

(c) Varying the number of observations

(d) Varying the noise magnitude

Figure 1 Comparison of prediction models (for linear regression).

4. Centralized: This is a naïve OLS model where we assume that for each feature, all the items have the same coefficient.

5. Clustering: We first cluster the items using k-means based on their features. We then fit an OLS model for each cluster.

As we can see from Figure 1, our algorithm outperforms all the benchmarks in all settings, in terms of out-of-sample R^2. Similar results were observed for the MSE. As expected, as we increase the number of observations, the prediction accuracy of the DAC algorithm quickly converges to 1. We find that the convergence is faster relative to the decentralized OLS method (with or without regularization), hence highlighting the benefit of the DAC algorithm under limited data availability. This clearly demonstrates the power of data aggregation and clustering present in our algorithm. Interestingly, the performance is not greatly affected by the number of items or by the number of features (indeed, the performance of each method depends on the proportion of the different feature types rather than the absolute number of features). Finally, as expected, a higher σ^2 negatively impacts the prediction accuracy of all methods, as it makes identifying the structure more challenging. Still, our proposed algorithm has a substantial advantage relative to the four benchmarks. To complement

Table 2 **Performance statistics over 100 independent trials (each value is the out-of-sample R^2). Parameters: $m = 20$, $d = 8$, $n = 100$, $\sigma^2 = 1$, $p = 2/3$, $q = 1/6$.**

Method	Min	Max	Mean	Std
DAC	0.691	0.943	0.876	0.038
Decentralized	0.564	0.924	0.825	0.061
Centralized	0.025	0.907	0.403	0.199
Clustering	0.056	0.907	0.405	0.197
Decentralized-Lasso	0.48	0.909	0.799	0.071

Table 3 **Performance across items (each value is the out-of-sample MSE). Same parameters as Table 2.**

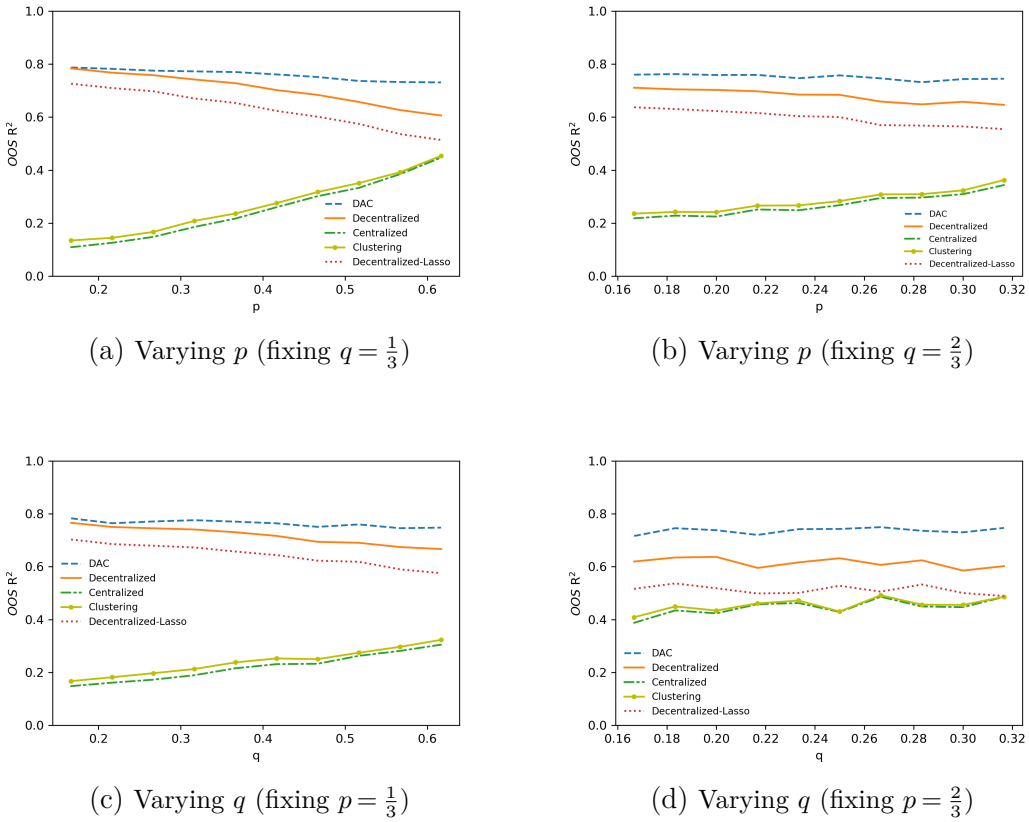
Method	Min	Max	Mean	Std
DAC	0.216	2.61	1.041	0.495
Decentralized	0.304	3.515	1.542	0.653
Centralized	0.386	13.25	3.725	3.02
Clustering	0.273	13.69	3.69	2.983
Decentralized-Lasso	0.285	6.728	1.784	1.07

the results of Figure 1, we report some statistics on the results of the different methods. In Table 2, we report the performance statistics (measured by the out-of-sample R^2) over the 100 independent trials. In Table 3, we report the performance statistics across $n = 100$ different items (in this case, the per-item performance is captured by the MSE). As we can see from both tables, the DAC algorithm outperforms all the benchmarks for all metrics. In addition, the standard deviation—both across instances and across items—reduces significantly, hence suggesting that our proposed algorithm also decreases the variability of the performance.

Figure 2 presents the performance of the methods when we vary the structure probability of the features in terms of aggregation level. When a large proportion of the features are at the department level (i.e., p is getting close to 1), all five methods perform well (the DAC algorithm still performs best in all cases). However, for instances where the structure is more diverse, our algorithm significantly outperforms the four benchmarks. Specifically, in all cases, the DAC algorithm leverages the structure of the problem by aggregating the data, ultimately yielding a higher prediction accuracy.

To further evaluate the performance of the DAC algorithm, we consider two additional performance metrics: measuring the capability to correctly identify the data aggregation levels of the features, and measuring the capability to accurately recover the cluster structure of the items with respect to cluster-level features. To quantify the first capability, we investigate the proportion of features for which the DAC algorithm correctly identifies their aggregation level. This metric is defined in a similar fashion as the *Accuracy* metric used by Jagabathula et al. (2018). To quantify the second capability, we measure the ability to recover the cluster structures using the *Rand index* (see Rand 1971), defined as

$$RI := \frac{a+b}{\binom{n}{2}},$$



(a) Varying p (fixing $q = \frac{1}{3}$)

(b) Varying p (fixing $q = \frac{2}{3}$)

(c) Varying q (fixing $p = \frac{1}{3}$)

(d) Varying q (fixing $p = \frac{2}{3}$)

Figure 2 Comparison of prediction models (for linear regression).

where a is the number of item pairs that belong to the same cluster and predicted to be in the same cluster, b is the number of item pairs that are in different clusters and predicted to be in different clusters, and $\binom{n}{2} = n(n-1)/2$ is total number of item pairs. The higher RI is, the more capable the algorithm is in recovering the true cluster structure. As a benchmark approach to recover the cluster structure, we consider the standard k-means clustering algorithm, where we first cluster the items into different clusters based on the average normalized values of the features $\boldsymbol{X}$, and then estimate a demand model for each cluster.

We highlight that to compute the two above metrics, one needs to have access to the ground-truth values of the aggregation levels and the cluster structure, which is not the case in settings based on real data. In such settings, the only way to evaluate the performance of our DAC algorithm is to compute the out-of-sample prediction accuracy as shown in Section 6.

We consider the same computational setting as in Table 1. We report the results of our simulation analyses in Figures 7–9 in Appendix D. As shown in Figure 7, the DAC algorithm can identify the data aggregation level of each feature with a reasonably high level of accuracy. Similarly, the DAC

algorithm can correctly recover the cluster structure in most instances.[3] In Figure 8, we conduct a comprehensive sensitivity analysis on both performance metrics with respect to changes in the different model parameters including the number of items n, the number of features d, the number of data observations m, the noise magnitude σ, and the number of clusters k (we assume that the number of clusters is the same for all cluster-level features). We find that the performance of our approach is robust with respect to the different parameters, generating a reasonable performance for all problem instances we examine. In particular, the DAC algorithm is most accurate in identifying aggregation levels and recovering the cluster structure when the noise magnitude is low (i.e., a small value of σ) and when the sample size m is large. This is inline with our theoretical result that the DAC algorithm is consistent in identifying aggregation levels and the cluster structure (i.e., Proposition 3(a)).

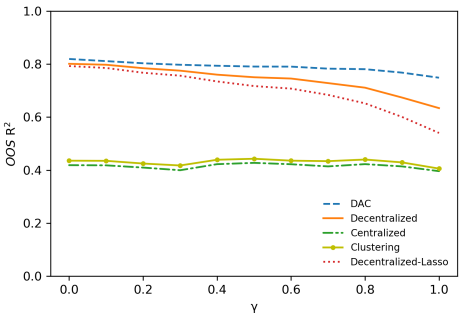
Finally, Figure 9 reports the comparison of the Rand index between the DAC algorithm and the benchmark clustering algorithm (based on k-means). Our simulation results show that the DAC algorithm outperforms the standard clustering approach in recovering the true cluster structure. Importantly, DAC dominates the benchmark with a sizable increase (15%–20%) in the Rand index.

5.2. Two Types of Items

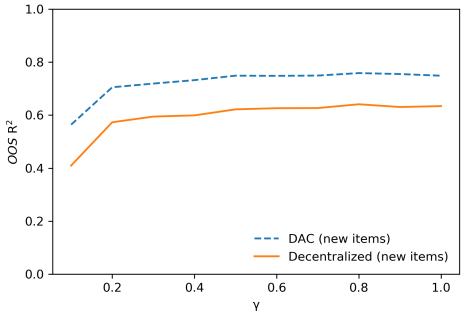
In this section, we consider an interesting setting where a subset of the items have limited data (referred to as 'new items'), whereas other items have abundant data (referred to as 'old items'). Our goal is to showcase that our proposed DAC algorithm can leverage the data from the old items to improve the prediction accuracy for *both* types of items. These computational experiments complement the analytical result derived in Proposition 6 by considering a non-asymptotic regime. Specifically, we consider a setting with $n = 20$ items, $d = 5$ features, $\sigma^2 = 1$, $p = 2/3$, $q = 1/6$, $m_{new} = 10$, and $m_{old} = 30$. It thus corresponds to the situation where the old items have three times more observations than the new items. We then vary the proportion of new items, captured by the parameter $\gamma \in [0, 1]$. When $\gamma = 0$, it corresponds to the setting where all the items have abundant data (i.e., they all have $m_{old} = 30$ observations). As before, we consider 100 independent trials and use the same fine-tuned values of θ, R_U, and R_L.

The results are presented in Figure 3. In the top panel, we show the average out-of-sample R^2 across all items as a function of γ. As before, the DAC algorithm consistently outperforms all four benchmarks. As expected, the benefit of the proposed algorithm relative to the decentralized OLS method increases as γ increases. In the bottom two panels, we compute the out-of-sample R^2 separately for new items (Panel b) and old items (Panel c) while focusing on the comparison between DAC and decentralized methods. The results readily confirm both insights drawn from Proposition 6: (i) the

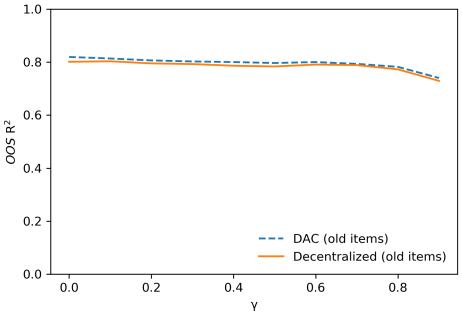
[3] We note that the notion of "good performance" in this context is subjective, as it depends on the performance of alternative methods. We will consider below a benchmark approach for the task of recovering the cluster structure and convey the superiority of our approach.



(a) All items



(b) New items only



(c) Old items only

Figure 3 Comparison of prediction models for a setting with two types of items (for linear regression).

DAC algorithm improves the prediction accuracy for both types of items, and (ii) the improvement is more substantial for the items with limited data.

5.3. Logistic Regression

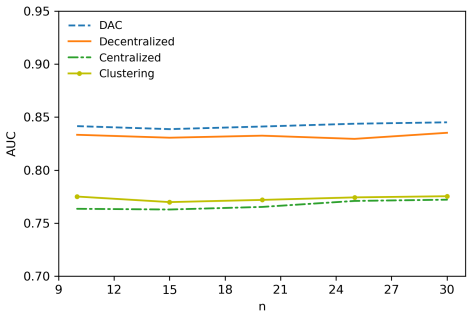
In this section, we present computational experiments for a classification problem in which the data-generating process is the logistic regression model, that is, for $i = 1, \ldots, n$ and $j = 1, \ldots, m$,

$$Y_{i,j} = \text{logit}\left(\sum_{l \in \mathcal{D}_s} X^l_{i,j}\beta^s_l + \sum_{l \in \mathcal{D}_n} X^l_{i,j}\beta^n_{i,l} + \sum_{l \in \mathcal{D}_c} X^l_{i,j}\beta^c_{\varsigma(i,l),l}\right) + \epsilon_{i,j},$$

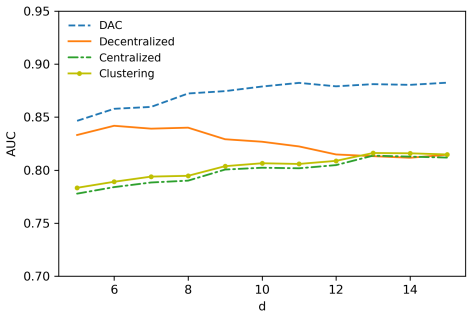
where $\text{logit(u)} := 1/(1+\exp(-u))$ is the Sigmoid function. We consider a similar setting as in Section 5.1 and use the same values of k, θ, R_U, and R_L as before. We first generate the data matrix $\boldsymbol{X}$ from a uniform $[0,1]$ distribution and the $\boldsymbol{\beta}$ coefficients from a uniform $[-5,5]$ distribution. The outcome variable Y is then generated based on a Bernoulli distribution with parameter $\mu := \text{logit}(\boldsymbol{X}\boldsymbol{\beta})$. As in the linear regression case, we systematically vary one parameter at a time. The parameters' value ranges are summarized in Table 4.

Table 4 Parameters used in Section 5.3.

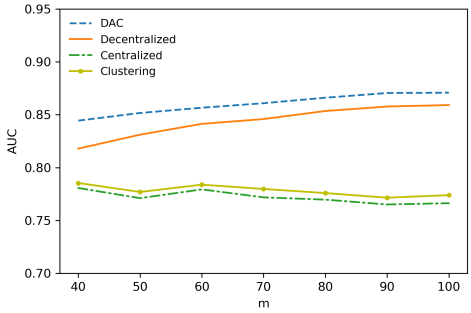
Parameter	Range of values
Number of items (n)	$[10, 30]$
Number of features (d)	$[5, 15]$
Number of observations (m)	$[40, 100]$



(a) Varying the number of items



(b) Varying the number of features



(c) Varying the number of observations

Figure 4 Comparison of prediction models (for logistic regression).

Following several prior studies on binary classification problems, we use the AUC as the metric to evaluate the performance of the different models. AUC is defined as the area under the receiver operating characteristic (ROC) curve (see, e.g., Bradley 1997). It can be interpreted as the probability that a prediction model is correctly ranking a random positive outcome higher than a random negative outcome. We compare our DAC algorithm relative to three benchmarks: decentralized, centralized, and clustering (the definition of each algorithm is similar to that in Section 5.1).[4] For each instance, we generate 100 independent trials and report the average out-of-sample AUC scores. As we can see from Figure 4, our method outperforms the benchmarks in all cases. Regardless of how we vary $\{n, m, d\}$, the DAC algorithm outperforms the three other methods in terms of prediction accuracy. Furthermore, a similar result as in Figure 2 was also observed for the logistic regression model (the details are omitted to avoid repetition).

To summarize, our simulation studies demonstrate a substantial and robust performance improvement for our proposed algorithm relative to several benchmarks (which are commonly used in practice

[4] Since estimating a decentralized model with ℓ_1 regularization is computationally prohibitive for the logistic regression setting, we only show the performance of the decentralized model without regularization.

and in the literature), even when the sample size is limited. Various other benchmarks will be considered in the next section, where the prediction algorithms are implemented. For both regression and classification problems, the DAC algorithm efficiently aggregates data and correctly identifies the data aggregation levels of the features and the cluster structures of the items, thus ultimately improving the prediction accuracy. In the next section, we apply the DAC algorithm using retail data to showcase its benefits in a practical setting.

6. Applying the DAC Algorithm to Retail Data

In this section, we apply the DAC algorithm to a retail dataset from a large global retailer (we cannot reveal the name of the retailer due to a non-disclosure agreement). We first provide a detailed description of the data and then test the prediction performance of the DAC algorithm relative to a broad range of benchmarks (we consider a total of 15 commonly used benchmarks). Finally, based on our computational findings, we draw managerial insights that can help retailers infer which features should be aggregated in practice.

6.1. Data

We have access to the retailer's online sales data. The dataset includes the weekly sales information of three departments between November 2013 and October 2016. A typical department comprises 100–150 SKUs. In addition to the weekly sales information, the dataset includes the weekly price, a promotion indicator (i.e., whether an item was promoted), the vendor, and the color of the SKU. Table 5 summarizes the specifics of each department. The size corresponds to the number of items in each department, and the numbers in parentheses are the standard deviations.

Table 5 Summary statistics of the data from each department.

Dept	Size	Observations	Weekly sales	Price	Promotion frequency	Discount rate
1	147	19,064	108.11 (377.45)	42.92 (38.80)	31.3%	4.2%
2	134	20,826	254.23 (517.07)	8.94 (8.67)	7.7%	6.4%
3	125	14,457	68.68 (691.55)	97.61 (67.12)	8.5%	1.5%

As we can see from Table 5, each department has a large number of observations (here, the observations are at the SKU-week level). There is also a great variation in terms of weekly sales, prices, promotion frequency, and discount rates across the different departments. Table 6 provides a brief description of the different fields in our dataset. The effective weekly price is computed as the total weekly revenue divided by the total weekly sales. Functionality is a segmentation hierarchy used by the firm to classify several SKUs from the same department into sub-categories.

Based on the features available in our data, we consider the following model specification:

$$\begin{aligned} Y_{i,t} = {} & \beta^i_{\text{Trend}} \cdot T_{i,t} + \beta^i_0 \cdot p_{i,t} + \beta^i_1 \cdot \text{PromoFlag}_{i,t} + \beta^i_2 \cdot \text{Fatigue}_{i,t} + \beta^i_3 \cdot \text{Seasonality}_{i,t} + \\ & + \beta^i_4 \cdot \text{Functionality}_{i,t} + \beta^i_5 \cdot \text{Color}_{i,t} + \beta^i_6 \cdot \text{Vendor}_{i,t} + \epsilon_{i,t}. \end{aligned} \tag{17}$$

Table 6 Fields in our dataset (observations are aggregated at the SKU-week level).

Fields	Description
SKU ID	Unique SKU ID
Week	Week index
Year	Year index
Units	Total weekly sales of a specific SKU
Price	Effective weekly price of a specific SKU
PromoFlag	Whether there was a promotion during that week
Functionality	Class index of a specific SKU
Color	Color of a specific SKU
Vendor	Vendor of a specific SKU

Equation (17) includes the following features: $Y_{i,t}$ is the total weekly sales of item i in week t (our dependent variable), $T_{i,t}$ is the trend variable of item i (we normalize the year so that $T_i = 0, 1, 2, 3$), $p_{i,t}$ is the effective price of item i in week t, $\text{PromoFlag}_{i,t}$ is a binary variable indicating whether there is a promotion for item i in week t, $\text{Fatigue}_{i,t}$ is the number of weeks since the most recent promotion for item i (if there is no previous promotion, $\text{Fatigue}_{i,t} = 0$; this feature allows us to capture the post-promotion dip effect), Seasonality is a categorical variable that measures the weekly or monthly effect on sales (we use one-hot encoding), and $\epsilon_{i,t}$ is an additive unobserved error. The remaining three variables are categorical variables indicating the web class index (functionality), color, and vendor of the SKU, respectively.

6.2. Prediction Performance

6.2.1. Benchmarks. As in Section 5, we compare the performance of the DAC algorithm relative to the same four benchmarks (decentralized, decentralized-Lasso, centralized, and clustering). In this section, for our analysis with real data, we also consider the following additional benchmarks: decentralized-elasticnet, DBSCAN clustering, OPTICS clustering, centralized-ISI (item-specific intercepts), centralized-K-means CFE (cluster-fixed effects), centralized-DBSCAN CFE, centralized-OPTICS CFE, decision tree, random forest, gradient boosted tree, and neural network.[5] Note that we consider three different clustering methods as benchmarks (k-means, DBSCAN, and OPTICS), as well as machine-learning based methods. Finally, we test adding fixed effects to the decentralized model under several configurations. We thus compare the DAC algorithm's performance to a total of 15 benchmarks, which are commonly used in both academia and practice.

It is important to mention that when implementing our DAC algorithm, we need to slightly adapt the clustering step of the algorithm. To ensure that we output a single cluster structure, we first collect the estimated coefficients from all cluster-level features and then fit a multi-dimensional k-means model (instead of a one-dimensional k-means for each feature). To avoid over-fitting, one can also include a ℓ_1-regularization term in the aggregated estimation (Step 7 of Algorithm 1).

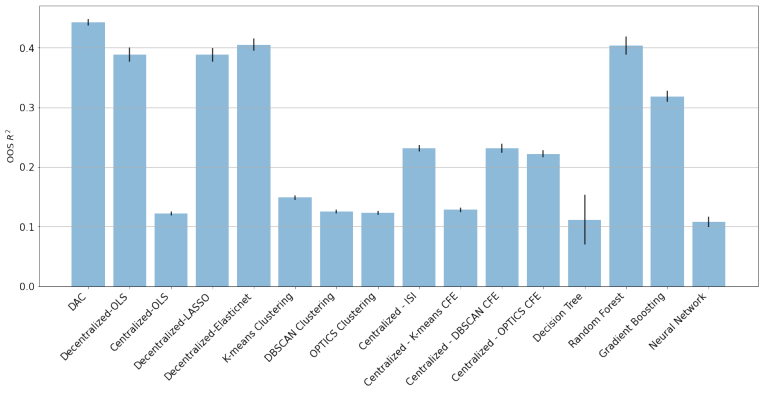
[5] The implementation details of the aforementioned methods are quite generic and are available upon request.

6.2.2. Implementation of DAC. Since the DAC algorithm has four hyper-parameters (k, θ, R_U, R_L), we use an extensive cross-validation procedure to fine-tune these parameters and select the best model. For each department, we first randomly split the data into training (70%) and testing (30%) sets. We assume that each design parameter lies within a pre-specified range: $k \in \{3, 4, \ldots, 10\}$, $\theta \in \{0.01, 0.05, 0.1, 0.5\}$, $R_U \in \{0.7, 0.8, 0.9\}$, and $R_L \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ (we add the option $\theta = 0.5$ to include the decentralized method as a special case). For each combination of hyper-parameters, we perform a five-fold cross-validation by fitting the model on 80% of the training data and compute the R^2 based on the remaining 20% of the data. This procedure is repeated five times for each parameter combination, and we compute the average R^2 over the five folds. We next select the best model based on the average cross-validation performance. Finally, the out-of-sample R^2 is computed on the testing set. Furthermore, since the train/cross-validation/test split is done randomly, we conduct 100 independent trials and report the mean and the 95%-confidence interval of the out-of-sample R^2.

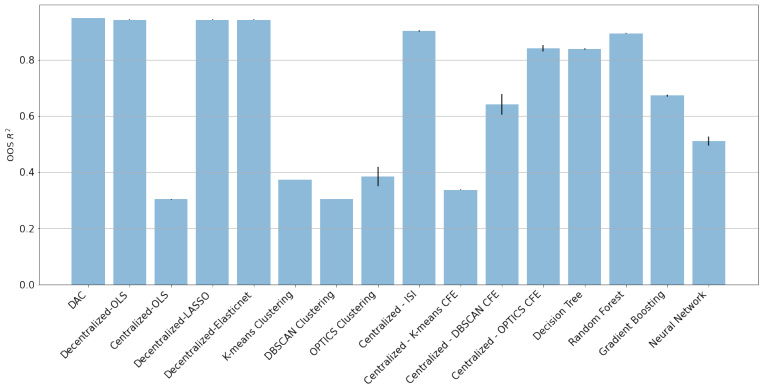
Our computational environment relies on the resources of Compute Canada and, more precisely, on the Volta generation Nvidia V100-SXM2-16Go node accessible on the Beluga computing network, with 6Go of memory (V100 offers the performance of up to 100 CPUs in a single GPU).

6.2.3. Results. We present the results for the out-of-sample R^2 using a five-fold cross validation in Figure 5. As expected, we obtain similar results for the MSE and the mean absolute percentage error (the details are omitted for conciseness). In Figure 5, each bar represents the average out-of-sample performance, and the length of the vertical line corresponds to the 95%-confidence interval across 100 trials. For all three departments, the DAC algorithm not only achieves a better average prediction performance but also often has a smaller variance. This shows that our algorithm is robust to different train/test splits, which is very desirable in practice. In addition, DAC outperforms all 15 benchmarks regardless of data quality. More precisely, Department 2 seems to have high-quality data and predictability power, whereas the data for Department 3 seems to be of lower quality (the number of observations per SKU is the smallest for Department 3, and data variability is high). Irrespective of the data quality, the DAC algorithm yields a clear improvement in prediction accuracy relative to all the benchmarks we considered. For completeness, we also recorded the in-sample R^2 values and consistently observed that the DAC algorithm reduces the amount of over-fitting relative to the other methods.

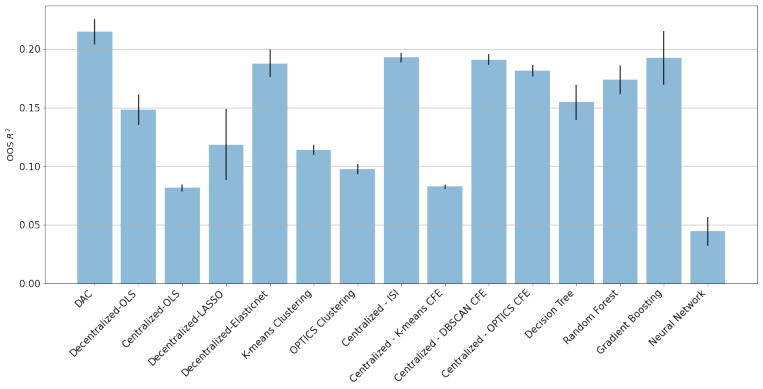
6.2.4. Time-Based Split Results. So far, we considered testing the different methods by using a random-based split. This allows us to perform a cross validation and, ultimately, compute confidence intervals in order to make statistical claims on the performance comparisons. An alternative way is to use a time-based split. Specifically, we can split the data by using the first $Y\%$ weeks for



(a) Department 1



(b) Department 2



(c) Department 3

Figure 5 **Performance comparison using real data (metric: out-of-sample R^2, random split).**

training (e.g., $Y = 70$) and the remaining $1 - Y\%$ for testing. This alternative data splitting rule is attractive in terms of prediction applicability but lacks statistical support in terms of comparing the different methods. When opting for a time-based split, two options are possible: an item-based split or an absolute split. In an item-based split, we divide the data into training and testing sets for each item separately. In an absolute split, we look at the total number of weeks in the dataset and then divide the data for all the items using the same week threshold. These two options differ

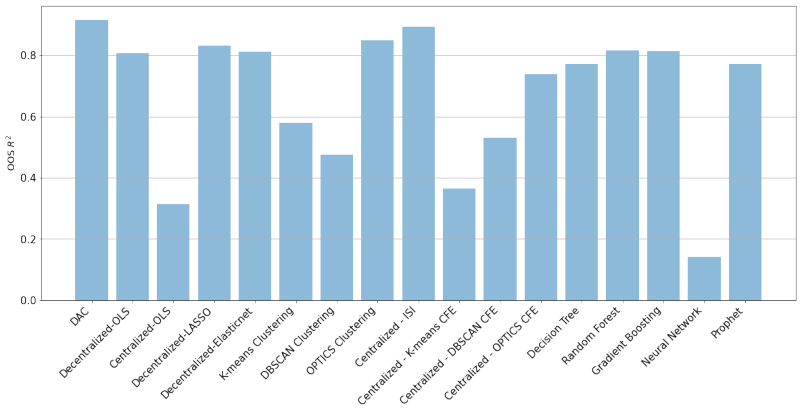


Figure 6 **Comparison using real data for Department 2 (metric: out-of-sample R^2, time-based split).**

Table 7 **Number of estimated coefficients.**

Department	Decentralized	DAC
1	9,408	6,095
2	6,298	4,977
3	3,000	892

when different items are introduced to the assortment at different times. In Figure 6, we present the results for Department 2 (the highest performing department) under an item-based split using 70% for training and 30% for testing. We obtained similar results for other split ratios and when using an absolute split. When using a time-based split, we can also consider time-series methods. Specifically, we include the Prophet method as an additional benchmark (Taylor and Letham 2018), which is considered among the state-of-the-art time-series approaches for demand prediction. Overall, as in the random-based split, the DAC algorithm outperforms all the benchmarks.

In summary, our results based on real data from three retail departments strongly suggest that the DAC algorithm has superior performance in terms of prediction accuracy. In addition to outperforming 15 benchmarks, our method reduces over-fitting and provides a great interpretability advantage. Specifically, it can help retailers identify the correct level of data aggregation for the different features and uncover the underlying cluster structure. Unlike other methods, the model output yields meaningful managerial insights, as we discuss next.

6.3. Managerial Insights

So far, we focused on the prediction performance of our proposed method. We then apply the DAC algorithm to our dataset and examine the estimation output. Our goal is to draw managerial insights into the hierarchical structure of the features. Next, we summarize our findings.

- The DAC algorithm can significantly reduce the model dimension. In Table 7, we report the number of estimated coefficients for the decentralized and DAC approaches across all three departments. Depending on the department, the number of estimated coefficients is reduced by 20% to 70%. The

results in Table 7 confirm that shared coefficients do occur in practice and that data aggregation can play an important role in correctly identifying the aggregation structure.

- Practitioners often argue that seasonality features should be aggregated at the department level for demand prediction (e.g., Cohen et al. 2017, Vakhutinsky et al. 2019). Using our retail dataset, we discover that this is indeed the case. If we model seasonality at the month level (i.e., we use 11 dummy variables for each calendar month), we find that for two departments, all 11 variables should indeed be estimated at the department level (whereas for the third department, the same is true for 7 out of 11 variables). If we instead model seasonality at the week level (i.e., we use 51 dummy variables for each week of the year), we find that 48 out of 51 variables should be estimated at the department level for two departments (and 18 out 51 for the third department). Thus, our findings validate and refine a well-known insight in practice.
- The price feature is unarguably one of the most important features for demand prediction in retail. According to our estimation results, for all three departments, we obtain a distinct coefficient for the price feature, implying that the price coefficient should be estimated at the SKU level. This typically holds for departments with a heterogeneous item collection.
- We find that the fatigue and promotion features should be estimated at the department level for all three departments (except the fatigue feature for one department). This is an interesting insight that can guide retailers when deciding their promotion strategy.
- We also find that all vendor and color dummy variables should be estimated at the SKU level. This is unsurprising given that most vendor-color combinations are unique for a specific SKU.
- The functionality dummy variables have different aggregation levels and cluster structures. Interestingly, most cluster-level features come from functionality. One possible explanation is that the functionality feature is obtained based on the hierarchy structure used by the company. Thus, SKUs with similar characteristics are usually labeled under the same functionality, making the cluster structure more prominent for functionality features. Retailers can use such results to potentially revise and improve their hierarchical structure and product segmentation.

7. Conclusion

Demand prediction or sales forecasting is an important task faced by most retailers. Improving prediction accuracy and drawing insights on data aggregation can significantly impact retailers' decisions and profits. When designing and estimating predictive models, retailers need to decide the aggregation level of each feature (e.g., seasonality and price). Some features may be estimated at the SKU level, others at the department level, and the rest at a cluster level. Traditionally, this problem was addressed by trial-and-error or by relying on past experience. It is common to see data scientists testing a multitude of model specifications until they find the best aggregation level for each feature.

Such an ad-hoc approach can be tedious and is not scalable for cases with a large number of features and items. The goal of this paper is to develop an efficient method to simultaneously determine (i) the correct aggregation level of each feature, (ii) the underlying cluster structure, and (iii) the estimated coefficients.

We propose a method referred to as the Data Aggregation with Clustering (DAC) algorithm. The DAC algorithm can determine the correct aggregation level and identify the cluster structure of the items. This method is tractable even when data dimensionality is high, and it can significantly improve the efficiency in estimating the model parameters. We first derive several analytical results to demonstrate the validity and benefits of our proposed method. Specifically, we show that the DAC algorithm yields a consistent estimate, along with improved asymptotic properties relative to the decentralized method. We then go beyond the theory and implement the DAC algorithm using a large retail dataset. In all our computational tests, we observe that the proposed method significantly improves prediction accuracy relative to a multitude of benchmarks. Finally, we convey that our method can help retailers uncover useful insights from their data.

Acknowledgments

We thank Paul-Emile Gras and Arthur Pentecoste who helped us conduct the computations presented in Sections 5 and 6. We also thank the retail partner for sharing data and Compute Canada for allowing us to use their computing resources. The second author is grateful for the financial support from the National Natural Science Foundation of China [NSFC71802133] and the Shanghai Eastern Scholar Program [QD2018053].

References

Baardman L, Levin I, Perakis G, Singhvi D (2017) Leveraging comparables for new product sales forecasting, available at SSRN 3086237.

Bernstein F, Modaresi S, Sauré D (2019) A dynamic clustering approach to data-driven assortment personalization. *Management Science* 65(5):2095–2115.

Bertsimas D, Kallus N (2020) From predictive to prescriptive analytics, management Science.

Bradley AP (1997) The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition* 30(7):1145–1159.

Caro F, Gallien J (2010) Inventory management of a fast-fashion retail network. *Operations Research* 58(2):257–273.

Cohen MA, Lee HL (2020) Designing the right global supply chain network. *Manufacturing & Service Operations Management* 22(1):15–24.

Cohen MC, Kalas JJ, Perakis G (2021) Promotion optimization for multiple items in supermarkets. *Management Science* 67(4):2340–2364.

Cohen MC, Leung NHZ, Panchamgam K, Perakis G, Smith A (2017) The impact of linear optimization on promotion planning. *Operations Research* 65(2):446–468.

Cooper LG, Baron P, Levy W, Swisher M, Gogos P (1999) Promocast™: A new forecasting method for promotion planning. *Marketing Science* 18(3):301–316.

Cooper LG, Giuffrida G (2000) Turning datamining into a management science tool: New algorithms and empirical results. *Management Science* 46(2):249–264.

Dekker M, Van Donselaar K, Ouwehand P (2004) How to use aggregation and combined forecasting to improve seasonal demand forecasts. *International Journal of Production Economics* 90(2):151–167.

Donti P, Amos B, Kolter JZ (2017) Task-based end-to-end model learning in stochastic optimization. *Advances in Neural Information Processing Systems*, 5484–5494.

Elmachtoub AN, Grigas P (2017) Smart" predict, then optimize", arXiv preprint arXiv:1710.08005.

Fahrmeir L, Kaufmann H, et al. (1985) Consistency and asymptotic normality of the maximum likelihood estimator in generalized linear models. *The Annals of Statistics* 13(1):342–368.

Feng Q, Shanthikumar JG (2018) How research in production and operations management may evolve in the era of big data. *Production and Operations Management* 27(9):1670–1684.

Ferreira KJ, Lee BHA, Simchi-Levi D (2016) Analytics for an online retailer: Demand forecasting and price optimization. *Manufacturing & Service Operations Management* 18(1):69–88.

Fildes R, Goodwin P, Önkal D (2019a) Use and misuse of information in supply chain forecasting of promotion effects. *International Journal of Forecasting* 35(1):144–156.

Fildes R, Ma S, Kolassa S (2019b) Retail forecasting: Research and practice, international Journal of Forecasting.

Greene WH (2003) *Econometric analysis* (Pearson Education India).

Gür Ali Ö (2013) Driver moderator method for retail sales prediction. *International Journal of Information Technology & Decision Making* 12(06):1261–1286.

Gür Ali Ö, Sayın S, Van Woensel T, Fransoo J (2009) Sku demand forecasting in the presence of promotions. *Expert Systems with Applications* 36(10):12340–12348.

Hastie T, Tibshirani R, Wainwright M (2019) *Statistical learning with sparsity: the lasso and generalizations* (Chapman and Hall/CRC).

Hu K, Acimovic J, Erize F, Thomas DJ, Van Mieghem JA (2019) Forecasting new product life cycle curves: Practical approach and empirical analysis. *M&SOM* 21(1):66–85.

Huang T, Fildes R, Soopramanien D (2014) The value of competitive information in forecasting fmcg retail product sales and the variable selection problem. *EJOR* 237(2):738–748.

Huang T, Fildes R, Soopramanien D (2019) Forecasting retailer product sales in the presence of structural change. *European Journal of Operational Research* 279(2):459–470.

Jagabathula S, Subramanian L, Venkataraman A (2018) A model-based embedding technique for segmenting customers. *Operations Research* 66(5):1247–1267.

Kesavan S, Gaur V, Raman A (2010) Do inventory and gross margin data improve sales forecasts for us public retailers? *Management Science* 56(9):1519–1533.

Kök AG, Fisher ML (2007) Demand estimation and assortment optimization under substitution: Methodology and application. *Operations Research* 55(6):1001–1021.

Li L, Lu Y, Zhou D (2017) Provably optimal algorithms for generalized linear contextual bandits. *International Conference on Machine Learning*, 2071–2080 (PMLR).

Liu S, He L, Max Shen ZJ (2021) On-time last-mile delivery: Order assignment with travel-time predictors. *Management Science* 67(7):4095–4119.

Ma S, Fildes R, Huang T (2016) Demand forecasting with high dimensional data: The case of sku retail sales forecasting with intra-and inter-category promotional information. *European Journal of Operational Research* 249(1):245–257.

Macé S, Neslin SA (2004) The determinants of pre-and postpromotion dips in sales of frequently purchased goods. *Journal of Marketing Research* 41(3):339–350.

MacQueen J, et al. (1967) Some methods for classification and analysis of multivariate observations. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1(14), 281–297.

McCullagh P, Nelder JA (2019) *Generalized Linear Models* (Routledge).

Park YW, Jiang Y, Klabjan D, Williams L (2017) Algorithms for generalized clusterwise linear regression. *INFORMS Journal on Computing* 29(2):301–317.

Ramdas A, Tibshirani RJ (2016) Fast and flexible admm algorithms for trend filtering. *Journal of Computational and Graphical Statistics* 25(3):839–858.

Rand WM (1971) Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association* 66(336):846–850.

Rigollet P (2015) 18. s997: High dimensional statistics lecture notes.

Shaffer JP (1995) Multiple hypothesis testing. *Annual Review of Psychology* 46(1):561–584.

Taylor SJ, Letham B (2018) Forecasting at scale. *The American Statistician* 72(1):37–45.

Tibshirani R (1996) Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)* 58(1):267–288.

Tibshirani R, Saunders M, Rosset S, Zhu J, Knight K (2005) Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67(1):91–108.

Tibshirani RJ, Taylor J (2011) The solution path of the generalized lasso. *The Annals of Statistics* 39(3):1335–1371.

Vakhutinsky A, Mihic K, Wu SM (2019) A prescriptive analytics approach to markdown pricing for an e-commerce retailer. *Journal of Pattern Recognition Research* 14(1):1–20.

Van Heerde HJ, Leeflang PS, Wittink DR (2000) The estimation of pre-and postpromotion dips with store-level scanner data. *Journal of Marketing Research* 37(3):383–395.

Zou H, Hastie T (2005) Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67(2):301–320.

Online Appendices to "Data Aggregation and Demand Prediction"

Maxime C. Cohen, Renyu Zhang, Kevin Jiao

Appendix A: Standard Generalized Linear Models

We next discuss the standard generalized linear models for completeness. Interested readers are referred to McCullagh and Nelder (2019) for a comprehensive presentation of the GLM theory. In particular, we will use the decentralized model to illustrate the classical likelihood theory of generalized linear models. More specifically, for item i, we assume that the conditional distribution of $Y_{i,j}$ given the features $\boldsymbol{X}_{i,j}$ comes from an exponential family with the following density:

$$\mathbb{P}\big(Y_{i,j}|\boldsymbol{X}_{i,j}\big)=\exp\left\{\frac{Y_{i,j}\boldsymbol{X}_{i,j}'\boldsymbol{\beta}_i-H(\boldsymbol{X}_{i,j}'\boldsymbol{\beta}_i)}{H_2(\gamma)}+H_3(Y_{i,j},\gamma)\right\}, \tag{18}$$

where $\gamma\in\mathbb{R}^+$ is a known scale parameter, and $H(\cdot)$, $H_2(\cdot)$, and $H_3(\cdot)$ are three real-valued normalization functions. The exponential family in Eq. (18) is very broad and includes Gaussian, binomial, Poisson, gamma, and inverse-Gaussian as special cases. It is straightforward to derive that, under the true parameter $\boldsymbol{\beta}_i$, the condition expectation of the outcome satisfies

$$\mathbb{E}\big[Y_{i,j}|\boldsymbol{X}_{i,j}\big]=H'(\boldsymbol{X}_{i,j}'\boldsymbol{\beta}_i)=G(\boldsymbol{X}_{i,j}'\boldsymbol{\beta}_i),$$

and the conditional variance of the outcome satisfies

$$\mathbb{V}\big(Y_{i,j}|\boldsymbol{X}_{i,j}\big)=H''(\boldsymbol{X}_{i,j}'\boldsymbol{\beta}_i)=G'(\boldsymbol{X}_{i,j}'\boldsymbol{\beta}_i)H_2(\gamma).$$

The log-likelihood function of parameter $\boldsymbol{b}_i$ for item i under model (18) is thus given by

$$\log\mathcal{L}_i(\boldsymbol{b}_i|\boldsymbol{Y}_i,\boldsymbol{X}_i)=\sum_{j=1}^{m}\left[\frac{Y_{i,j}\boldsymbol{X}_{i,j}'\boldsymbol{b}_i-H(\boldsymbol{X}_{i,j}'\boldsymbol{b}_i)}{H_2(\gamma)}+H_3(Y_{i,j},\gamma)\right]=\frac{1}{H_2(\gamma)}\cdot\sum_{j=1}^{m}\big[Y_{i,j}\boldsymbol{X}_{i,j}'\boldsymbol{b}_i-H(\boldsymbol{X}_{i,j}'\boldsymbol{b}_i)\big]+\text{constant},$$

where the constant is independent of the parameter $\boldsymbol{b}_i$. Therefore, the decentralized MLE $\hat{\boldsymbol{b}}_i$ is given by Eq. (3), which is equivalent to an iterative weighted least-squares procedure. The statistical theory of GLM and MLE establishes the asymptotic and finite sample properties of the decentralized estimator $\hat{\boldsymbol{b}}_i$. See Proposition 1 for more details.

Appendix B: Two Potential Methods

In this section, we introduce two potential methods to estimate the model in Eq. (1) and predict the demand, as well as discuss why these methods are not applicable to our setting.

B.1. Generalized ℓ_1-Regularized MLE

The first potential method we consider is the *generalized ℓ_1-/lasso-regularized MLE* (see, e.g., Tibshirani 1996, Tibshirani and Taylor 2011, Hastie et al. 2019) to estimate the coefficients. This approach revises the standard MLE by adding a generalized ℓ_1-regularizer. More specifically, the ℓ_1-regularized log-likelihood function of the aggregate model is given by:

$$\frac{1}{m}\sum_{j=1}^{m}\sum_{i=1}^{n}\big[Y_{i,j}\boldsymbol{X}_{i,j}'\boldsymbol{\beta}_i-H(\boldsymbol{X}_{i,j}'\boldsymbol{\beta}_i)\big]-\lambda\left(\sum_{i\neq i'}\sum_{l=1}^{d}|\beta_{i,l}-\beta_{i',l}|\right),\ \lambda>0, \tag{19}$$

where $H(\cdot)$ is the normalization mapping that satisfies $H'(u) = G(u)$ (see Appendix A). A canonical result in the statistics literature shows that ℓ_1-regularization will shrink the regularized terms to 0 and, thus, generate sparse solutions (see, e.g., Tibshirani 1996, Tibshirani and Taylor 2011, Zou and Hastie 2005). As a result, adding a generalized ℓ_1-regularizer to MLE may potentially be helpful to capture the fact that a feature at the aggregate or cluster level shares the same coefficient for different items. We note that ℓ_1-regularized MLE is in a similar spirit to the *fused lasso regression* (see, e.g., Tibshirani and Taylor 2011, Tibshirani et al. 2005). In the retail demand forecasting literature, Huang et al. (2014) and Ma et al. (2016) develop Lasso-based methodological frameworks to overcome the problem of the ultra-high dimensionality of the feature space under multiple product categories.

As shown by Tibshirani and Taylor (2011) and Ramdas and Tibshirani (2016), generating the ℓ_1-regularized MLE typically involves solving the dual problem multiple times along the solution path. Given the high-dimensional nature of the convex optimization problem in Eq. (19) (i.e., the number of decision variables is nd, which is at the magnitude of thousand or more in practice), estimating the coefficients is computationally prohibitive even for a linear regression specification (i.e., $G(u) = u$) as it involves inverting $(nd) \times (nd)$-matrices in each step to construct the solution path (see, e.g., Tibshirani and Taylor 2011, Ramdas and Tibshirani 2016). Therefore, though theoretically plausible, using the ℓ_1-regularized MLE is not tractable for our problem in practical settings.

B.2. Direct Optimization

We next consider the direct optimization approach that directly formulates the problem as a nonlinear program to jointly estimate the data aggregation levels, cluster structures, and feature coefficients. Since the data aggregation levels and cluster structures are unknown apriori, we need to use one-hot encoding to represent the aggregation levels and cluster structures. More specifically, we use $\delta^s_{i,l}$ to denote the indicator variable for feature l of item i to be at the aggregate level, $\delta^n_{i,l}$ to denote the indicator variable for feature l of item i to be at the individual level, and $\delta^c_{i,l,\varsigma}$ to denote the indicator variable for feature l of item i to be at the cluster level and item i being in cluster $\mathcal{C}_{l,\varsigma}$. Thus, there is a total of $2nd + n\sum_{l=1}^{d} k_l$ binary decision variables. For expositional convenience, we consider the linear regression model (i.e., $G(u) = u$). Then, the mean squared loss minimization can be written as:

$$\min_{\boldsymbol{\beta},\boldsymbol{\delta}} \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \left(Y_{i,j} - \boldsymbol{X}'_{i,j}\boldsymbol{\beta}_i\right)^2.$$

The constraints are not straightforward, so we next list them one by one. First, the δ variables are binary:

$$\delta^s_{i,l} \in \{0,1\},\ \delta^n_{i,l} \in \{0,1\},\ \text{and } \delta^c_{i,l,\varsigma} \in \{0,1\},\ \text{for all } 1 \le i \le n, 1 \le l \le d, 1 \le \varsigma \le k_l. \tag{20}$$

Second, a feature can be at one (and only one) data aggregation level, that is,

$$\delta^s_{i,l} + \delta^n_{i,l} + \sum_{\varsigma=1}^{k_l} \delta^c_{i,l,\varsigma} = 1,\ \text{for all } 1 \le i \le n, 1 \le l \le d. \tag{21}$$

Third, an aggregate-level feature should be at the aggregate level for all items:

$$\delta^s_{i,l} = \delta^s_{i',l},\ \text{for all } i \ne i', 1 \le l \le d. \tag{22}$$

Similarly, a SKU-level feature should be at the aggregate level for all items:

$$\delta^n_{i,l} = \delta^n_{i',l}, \text{ for all } i \neq i', 1 \leq l \leq d. \tag{23}$$

A cluster-level feature should be at the cluster level for all items:

$$\sum_{\varsigma=1}^{k_l} \delta^c_{i,l,\varsigma} = \sum_{\varsigma=1}^{k_l} \delta^c_{i',l,\varsigma}, \text{ for all } i \neq i', 1 \leq l \leq d. \tag{24}$$

The number of items in each cluster with respect to each cluster-level feature is at least two:

$$\sum_{i=1}^{n} \delta^c_{i,l,\varsigma} \geq \frac{2}{n} \sum_{\varsigma'=1}^{k_l} \sum_{i=1}^{n} \delta^c_{i,l,\varsigma'}, \text{ for all } 1 \leq l \leq d \text{ and } 1 \leq \varsigma \leq k_l. \tag{25}$$

For an aggregate-level feature, its coefficient should be the same for all items:

$$-2\bar{\beta}(1-\delta^a_{i,l}) \leq \beta_{i,l} - \beta_{i',l} \leq 2\bar{\beta}(1-\delta^a_{i,l}), \text{ for all } i \neq i' \text{ and } 1 \leq l \leq d, \tag{26}$$

where $\bar{\beta}$ is the maximum possible absolute value of the coefficients. Analogously, for a cluster-level feature and the items within the same cluster with respect to this feature, the coefficient should be identical:

$$-2\bar{\beta}(2-\delta^c_{i,l,\varsigma}-\delta^c_{i',l,\varsigma}) \leq \beta_{i,l} - \beta_{i',l} \leq 2\bar{\beta}(2-\delta^c_{i,l,\varsigma}-\delta^c_{i',l,\varsigma}), \text{ for all } i \neq i', \ 1 \leq l \leq d, \text{ and } 1 \leq \varsigma \leq k_l. \tag{27}$$

Based on (20), (21), (22), (23), (24), (25), (26), and (27), we formulate the direct optimization approach as the following mixed-integer second-order conic program (SOCP):

$$\begin{aligned}
\min_{\boldsymbol{\beta},\boldsymbol{\delta}} \quad & \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \left(Y_{i,j} - \boldsymbol{X}'_{i,j}\boldsymbol{\beta}_i\right)^2 \\
\text{s.t.} \quad & -\bar{\beta} \leq \beta_{i,l} \leq \bar{\beta} \ i = 1,2,\ldots,n, \ l = 1,2,\ldots,d, \\
& \delta^s_{i,l} \in \{0,1\}, \ \delta^n_{i,l} \in \{0,1\}, \text{ and } \delta^c_{i,l,\varsigma} \in \{0,1\}, \text{ for all } 1 \leq i \leq n, 1 \leq l \leq d, 1 \leq \varsigma \leq k_l, \\
& \delta^s_{i,l} + \delta^n_{i,l} + \sum_{\varsigma=1}^{k_l} \delta^c_{i,l,\varsigma} = 1, \text{ for all } 1 \leq i \leq n, 1 \leq l \leq d, \\
& \delta^s_{i,l} = \delta^s_{i',l}, \text{ for all } i \neq i', 1 \leq l \leq d, \\
& \delta^n_{i,l} = \delta^n_{i',l}, \text{ for all } i \neq i', 1 \leq l \leq d, \\
& \sum_{\varsigma=1}^{k_l} \delta^c_{i,l,\varsigma} = \sum_{\varsigma=1}^{k_l} \delta^c_{i',l,\varsigma}, \text{ for all } i \neq i', 1 \leq l \leq d, \\
& \sum_{i=1}^{n} \delta^c_{i,l,\varsigma} \geq \frac{2}{n} \sum_{\varsigma'=1}^{k_l} \sum_{i=1}^{n} \delta^c_{i,l,\varsigma'}, \text{ for all } 1 \leq l \leq d \text{ and } 1 \leq \varsigma \leq k_l, \\
& -2\bar{\beta}(1-\delta^a_{i,l}) \leq \beta_{i,l} - \beta_{i',l} \leq 2\bar{\beta}(1-\delta^a_{i,l}), \text{ for all } i \neq i' \text{ and } 1 \leq l \leq d, \\
& -2\bar{\beta}(2-\delta^c_{i,l,\varsigma}-\delta^c_{i',l,\varsigma}) \leq \beta_{i,l} - \beta_{i',l} \leq 2\bar{\beta}(2-\delta^c_{i,l,\varsigma}-\delta^c_{i',l,\varsigma}), \text{ for all } i \neq i', \ 1 \leq l \leq d, \text{ and } 1 \leq \varsigma \leq k_l.
\end{aligned} \tag{28}$$

We note that the mixed-integer SOCP in Eq. (28) has nd continuous decision variables, $2nd + n\sum_{l=1}^{d} k_l$ binary decision variables, and $O\left(n^2 d\left(\sum_{l=1}^{d} k_l\right)\right)$ linear constraints, which is intractable for a practical problem of a reasonable size.

A similar generalized clusterwise linear regression (CLR) model has been proposed by Park et al. (2017) to address a special case of our problem where all the features are at the cluster level and the cluster structure

is the same across all features. Park et al. (2017) show that the generalized CLR is NP-hard and propose column generation and metaheuristic genetic algorithms to solve this problem. Since our problem in Eq. (28) is more general with unknown data aggregation levels, the estimation methods proposed by Park et al. (2017) are not applicable and tractable in our setting.

Alternatively, one may solve problem (28) via a procedure that iteratively estimates the continuous coefficients and the binary decision variables for aggregation levels and cluster structures (in a similar way as in Baardman et al. 2017). The iterative procedure will stop once the binary variables remain the same for two consecutive iterations. Baardman et al. (2017) address the demand prediction problem when there are only cluster-level features (i.e., no aggregate-level and no SKU-level features). In their setting, this iterative procedure was proved to converge to the true coefficients and cluster structure (i.e., the estimate is consistent). In our setting, however, we cannot guarantee the consistency of the iterative optimization approach due to the heterogeneous data aggregation levels and the unknown cluster structures in our model. As a result, the iterative procedure is not a viable approach to solve problem (28) and estimate our model.

In conclusion, both the generalized ℓ_1-regularized MLE and the direct optimization approaches cannot be used to solve our problem in practice.

Appendix C: Proofs of Statements

We next provide the proofs of all the technical results.

Proof of Proposition 1

Part (a). The proof of the consistency results follows from a standard result in statistics stating that the maximum-likelihood estimator (MLE) is consistent under some regularity conditions that are satisfied by a generalized linear model. See, for example, Fahrmeir et al. (1985) and McCullagh and Nelder (2019).

Part (b). We first show the asymptotic normality in Eq. (5). This is a standard result in the MLE literature, which follows directly from, e.g., Theorem 3 of Fahrmeir et al. (1985).

We next prove the finite-sample normality result in Eq. (4). Since the smallest eigenvalue of $\Sigma_i = \mathbb{E}\left[\boldsymbol{X_{i,j}}\boldsymbol{X'_{i,j}}\right]$, $\lambda_{\min}(\Sigma_i)$, is strictly positive, by Proposition 1 of Li et al. (2017), there exists a (sufficiently large) threshold $\mathfrak{m}'_i$ such that, as long $m \geq \mathfrak{m}'_i$, the smallest eigenvalue of $\hat{\boldsymbol{V}}_i(m) := \sum_{j=1}^{m} \boldsymbol{X}_{i,j}\boldsymbol{X}'_{i,j}$, $\lambda_{\min}(\hat{\boldsymbol{V}}_i(m))$, can be arbitrarily large as m increases to infinity. Therefore, the condition of Theorem 1 in Li et al. (2017) (i.e., Equation (4) thereof) is satisfied.

We define $\boldsymbol{x} \in \mathbb{R}^d$ with $x_l = 1$ and all other $x_{l'} = 0$. Thus, $\boldsymbol{x}'(\hat{\boldsymbol{b}}(m)_i - \boldsymbol{\beta}_i) = \hat{b}_{i,l}(m) - \beta_{i,l}$ and the ℓ_2-norm of x associated with $\hat{V}_i(m)$ is

$$||\boldsymbol{x}||_{\hat{\boldsymbol{V}}_i(m)^{-1}} = \sqrt{\boldsymbol{x}'\hat{\boldsymbol{V}}_i(m)^{-1}\boldsymbol{x}} = \sqrt{(\hat{\boldsymbol{V}}_i(m)^{-1})_{l,l}} = \sqrt{\frac{1}{m}\left(\left(\frac{1}{m}\sum_{j=1}^{m}\boldsymbol{X}_{i,j}\boldsymbol{X}'_{i,j}\right)^{-1}\right)_{l,l}} \leq \sqrt{\frac{2(\Sigma_i^{-1})_{l,l}}{m}}, \tag{29}$$

when $m \geq \mathfrak{m}'_{i,l}$ for some threshold $\mathfrak{m}'_{i,l}$ by the strong law of large numbers. For any $\epsilon > 0$, we define

$$\delta := \exp\left(-\psi_{i,l}\cdot\epsilon^2\cdot m\right) \quad \text{where } \psi_{i,l} := \frac{\underline{g}_i^2}{18(\Sigma_i^{-1})_{l,l}\sigma^2}, \text{ and } \underline{g}_i := \inf\left\{G'(\boldsymbol{z}'\boldsymbol{b}_i) : \boldsymbol{z}\in\mathbb{R}^d, ||\boldsymbol{z}||\leq 1, ||\boldsymbol{b}_i - \boldsymbol{\beta}_i|| \leq 1\right\} > 0.$$

Hence, δ satisfies

$$\frac{3\sigma}{\underline{g}_i}\cdot\sqrt{\log\left(\frac{1}{\delta}\right)}\cdot\sqrt{\frac{2(\Sigma_i^{-1})_{l,l}}{m}} = \epsilon.$$

Thus, if $m \geq \mathfrak{m}_{i,l} := \max\{\mathfrak{m}'_i, \mathfrak{m}'_{i,l}\}$, with probability at least $1-3\delta$, the following inequality holds:

$$|\hat{b}_{i,l}(m) - \beta_{i,l}| = |\boldsymbol{x}'(\hat{\boldsymbol{b}}(m)_i - \boldsymbol{\beta}_i)| \leq \frac{3\sigma}{\underline{g}_i} \cdot \sqrt{\log\left(\frac{1}{\delta}\right)} \cdot ||\boldsymbol{x}||_{\hat{\boldsymbol{V}}_i(m)^{-1}} \leq \frac{3\sigma}{\underline{g}_i} \cdot \sqrt{\log\left(\frac{1}{\delta}\right)} \cdot \sqrt{\frac{2(\Sigma_i^{-1})_{l,l}}{m}} = \epsilon, \quad (30)$$

where the first inequality follows from Theorem 1 in Li et al. (2017) and the second from Eq. (29). Inequality (30) immediately implies that, for $m \geq \mathfrak{m}_{i,l}$, inequality (4) holds, hence proving Proposition 1. □

Proof of Proposition 2

The proof is similar to the proof of Proposition 1, so we only sketch it for brevity. We note that we can reformulate the aggregate model as a new GLM with $m \times n$ observations. We denote the outcome vector as $\tilde{\boldsymbol{Y}} \in \mathbb{R}^{mn}$. The feature design matrix $\tilde{\boldsymbol{X}}$ has $m \times n$ rows (representing observations) and d_x columns (representing the total number of features):

$$\tilde{Y}_j = G\left(\sum_{l=1}^{d_x} \tilde{X}_j^l \tilde{\beta}_l\right) + \epsilon_j, \quad (31)$$

where $\tilde{\beta}_l$ is the coefficient for feature l in the aggregate model and ϵ_j is the independent sub-Gaussian error term. With the new formulation in Eq. (31), the aggregate model can be viewed as a decentralized model with one item, d_x features, and $m \times n$ observations.

For **part (a)**, the proof follows directly from the standard MLE theory. For **part (b)**, both the finite sample and asymptotic normality follow from a similar argument as in the proof of Proposition 1. Finally, if $\mathcal{D}_c \cup \mathcal{D}_s \neq \emptyset$, then $\sqrt{m}b^a_{i,l}(m) = \sqrt{m}b^a_{i',l}(m)$ for $i \neq i'$ and $\mathcal{C}(i,l) = \mathcal{C}(i',l)$. In this case, the asymptotic distribution of $\sqrt{m}(\boldsymbol{b}^a(m) - \boldsymbol{\beta})$ is clearly degenerate, because it has some equal coordinates for all m. This concludes the proof of Proposition 2. □

Proof of Proposition 3

Part (a). We first show the inequality in Eq. (7).

- *Step 1.* If $\beta_{1,l} \neq \beta_{i,l}$ for some $i \neq 1$, then $\lim_{m\uparrow+\infty} \mathbb{P}\left[H^l_{1,i} \text{ is not rejected}\right] = 0$. This implies that the probability of Type-II error to falsely identify two different coefficients to be the same converges to 0.

We now assume that $\beta_{1,l} \neq \beta_{i,l}$ and use the notation $H^l_{1,i}(m)$ to make the dependence of $H^l_{1,i}$ on the sample size m explicit. The probability that $H^l_{1,i}(m)$ is not rejected (resp. is rejected) is denoted by $\mathfrak{p}(m)$ (resp. $\mathfrak{q}(m) := 1 - \mathfrak{p}(m)$). Since $\sqrt{m}\hat{b}_{1,l}(m)$ and $\sqrt{m}\hat{b}_{i,l}(m)$ are asymptotically normally distributed, there exists a constant $c_{i,l} > 0$ independent of m such that $H^l_{1,i}(m)$ is not rejected if and only if

$$|\sqrt{m}\hat{b}_{1,l}(m) - \sqrt{m}\hat{b}_{i,l}(m)| \leq c_{i,l}, \text{ for } m \text{ sufficiently large.}$$

We define $\varepsilon := \frac{1}{3} \cdot |\beta_{1,l} - \beta_{i,l}| > 0$. We assume that $m > \left(\frac{c_{i,l}}{\varepsilon}\right)^2$, i.e., $\frac{c_{i,l}}{\sqrt{m}} < \varepsilon$ and consider the case where $|\hat{b}_{1,l}(m) - \beta_{1,l}| \leq \varepsilon$ and $|\hat{b}_{i,l}(m) - \beta_{i,l}| \leq \varepsilon$. In this case,

$$|\hat{b}_{1,l}(m) - \hat{b}_{i,l}(m)| \geq \frac{1}{3}|\beta_{1,l} - \beta_{i,l}| = \varepsilon > \frac{c_{i,l}}{\sqrt{m}}.$$

Therefore, if $|\hat{b}_{1,l}(m)-\beta_{1,l}|\le\varepsilon$ and $|\hat{b}_{i,l}(m)-\beta_{i,l}|\le\varepsilon$, $H^l_{1,i}(m)$ will be rejected, which implies that if m is sufficiently large, we have

$$\begin{aligned}\mathfrak{q}(m)\ge&\mathbb{P}\left[|\hat{b}_{1,l}(m)-\beta_{1,l}|\le\varepsilon,|\hat{b}_{i,l}(m)-\beta_{i,l}|\le\varepsilon\right]\\=&1-\mathbb{P}\left[|\hat{b}_{1,l}(m)-\beta_{1,l}|>\varepsilon\text{ or }|\hat{b}_{i,l}(m)-\beta_{i,l}|>\varepsilon\right]\\\ge&1-\mathbb{P}\left[|\hat{b}_{1,l}(m)-\beta_{1,l}|>\varepsilon\right]-\mathbb{P}\left[|\hat{b}_{i,l}(m)-\beta_{i,l}|>\varepsilon\right]\\\ge&1-3\exp(-\psi_{1,l}\varepsilon^2m)-3\exp(-\psi_{i,l}\varepsilon^2m),\end{aligned}\tag{32}$$

where the second inequality follows from the union bound and the last inequality from Eq. (4). Inequality (32) implies that $\lim_{m\uparrow+\infty}\mathfrak{q}(m)=1$, or equivalently, $\lim_{m\uparrow+\infty}\mathfrak{p}(m)=0$, which proves *Step 1*. This also implies that, as $m\uparrow+\infty$, the probability that the DAC_α algorithm mis-specifies an individual-level feature as a cluster- or aggregate- level one, or a cluster-level feature as an aggregate-level one will shrink to 0 exponentially fast.

- *Step 2.* If $l\in\mathcal{D}_c$, then Step 6 of Algorithm 1 will produce a consistent estimate of the cluster structure with respect to feature l, that is,

$$\lim_{m\uparrow+\infty}\mathbb{P}\left[(\hat{\mathcal{C}}_{1,l},\hat{\mathcal{C}}_{2,l},\ldots,\hat{\mathcal{C}}_{k_l,l})\text{ is a permutation of }(\mathcal{C}_{1,l},\mathcal{C}_{2,l},\ldots,\mathcal{C}_{k_l,l})\right]=0.$$

We now fix feature $l\in\mathcal{D}_c$. We note that, for any $i\in\mathcal{C}_\varsigma$ $(1\le\varsigma\le k_l)$, the coefficient of feature l is $\beta^c_{\varsigma,l}$ and $\hat{b}_{i,l}$ converges to $\beta^c_{\varsigma,l}$ with a probability that exponentially decays in the sample size m. Thus, for the k-means algorithm $(k=k_l)$ applied to $\{\hat{b}_{1,l},\hat{b}_{2,l},...,\hat{b}_{n,l}\}$, the centers of the k_l clusters $\{\hat{c}_{1,l},\hat{c}_{2,l},...,\hat{c}_{k_l,l}\}$, where $\hat{c}_\varsigma$ is the center of cluster $\mathcal{C}_\varsigma$, will converge to the true coefficient vectors for cluster-level features $\beta^c_{\varsigma,l}$ up to a permutation on $\{1,2,...,k_l\}$. For notational convenience, we assume that $\hat{c}_{\varsigma,l}$ converges to $\beta^c_{\varsigma,l}$ for $1\le\varsigma\le k_l$.

If there is an item $i\in\mathcal{C}_{\varsigma,l}$ that is "mis-clustered" into $\hat{\mathcal{C}}_{\varsigma',l}$, we have, as $m\uparrow+\infty$, $\hat{b}_{i,l_1}$ converges to $\beta^c_{\varsigma,l}\ne\beta^c_{\varsigma',l}$, that is, for m sufficiently large,

$$|\hat{b}_{i,l}-\beta^c_{\varsigma,l}|<|\hat{b}_{i,l}-\beta^c_{\varsigma',l}|.$$

This implies that, for m sufficiently large,

$$|\hat{b}_{i,l}-\hat{c}_{\varsigma,l}|<|\hat{b}_{i,l}-\hat{c}_{\varsigma',l}|,$$

which contradicts the assumption that item $i\in\mathcal{C}_{\varsigma,l}$ is mis-clustered into $\hat{\mathcal{C}}_{\varsigma',l}$ and hence concludes the proof of *Step 2*. This also implies that, if $m\uparrow+\infty$, as long as a cluster-level feature is correctly specified, the cluster structure can also be correctly identified with probability 1.

- *Step 3.* Given any significance level $\alpha\in(0,1)$, the probability that the DAC_α algorithm mis-specifies any cluster-level feature as an individual one, or any aggregate-level feature as a cluster-level one or an individual-level one is upper bounded by $p(\alpha)>0$ as $m\uparrow+\infty$.

We first note that the probability that the DAC_α algorithm mis-specifies any cluster-level feature as an individual one, or any aggregate-level feature as a cluster-level one or an individual-level one is upper bounded by the probability of the event that all the features are at the aggregate level but the algorithm mis-specifies some feature to be at the cluster level or the individual level. We define the latter probability as $p(\alpha)$, which is the probability that $H^l_{1,i}$ is rejected for at least one (i,l) under $2\le i\le n$ and $1\le l\le d$ under the condition

that all the features are at the aggregate level. For the rest of the proof of *Step 3*, we assume that all features are at the aggregate level

We next quantify $p(\alpha)$ using multiple hypothesis testing (MHT) in the asymptotic regime ($m \uparrow +\infty$). With a slight abuse of notation, we use $\hat{\boldsymbol{b}} \in \mathbb{R}^{nd}$ to denote a virtual estimator following the same distribution as the limiting distribution (i.e., $m \uparrow +\infty$) of $\sqrt{m}(\hat{\boldsymbol{b}}(m) - \boldsymbol{\beta})$. By Proposition 1(b), $\hat{\boldsymbol{b}}$ follows a zero-mean multivariate normal distribution with covariance matrix $\mathcal{V} := \text{diag}(\mathcal{I}_1(\boldsymbol{\beta}_1)^{-1}, \mathcal{I}_2(\boldsymbol{\beta}_2)^{-1}, ..., \mathcal{I}_n(\boldsymbol{\beta}_n)^{-1})$, which is block diagonal. We define an $(n-1)d$-by-nd matrix $\mathcal{T}$ such that $\hat{\boldsymbol{t}} := \mathcal{T} \cdot \hat{\boldsymbol{b}}$ is the joint estimator for Step 6 (Hypothesis Testing) of Algorithm 1, that is,

$$\hat{\boldsymbol{t}} := \mathcal{T} \cdot \hat{\boldsymbol{b}} = \begin{pmatrix} \hat{b}_{1,1} - \hat{b}_{2,1} \\ \hat{b}_{1,1} - \hat{b}_{3,1} \\ \cdots \\ \hat{b}_{1,1} - \hat{b}_{n,1} \\ \cdots \\ \hat{b}_{1,d} - \hat{b}_{2,d} \\ \hat{b}_{1,d} - \hat{b}_{3,d} \\ \cdots \\ \hat{b}_{1,d} - \hat{b}_{n,d} \end{pmatrix} \in \mathbb{R}^{(n-1)d}.$$

Therefore, $\hat{\boldsymbol{t}}$ is normally distributed with mean $\mathbf{0} \in \mathbb{R}^{(n-1)d}$ and covariance matrix $\tilde{\mathcal{V}} := \mathcal{T} \cdot \mathcal{V} \cdot \mathcal{T}'$. Then, in Algorithm 1, $\mathcal{H}^l_{1,i}$ ($1 \le l \le d$ and $2 \le i \le n$) is rejected if and only if $\tilde{t}_{(n-1)(l-1)+i-1} = \hat{b}_{1,l} - \hat{b}_{i,l}$ is located outside the interval

$$\mathbb{I}_j(\alpha) := \left[-\mathcal{V}_{j,j} \Phi^{-1}\left(1 - \frac{\alpha}{2}\right), \mathcal{V}_{j,j} \Phi^{-1}\left(1 - \frac{\alpha}{2}\right) \right], \text{ where } j := (n-1)(l-1) + i - 1 \text{ and } \Phi^{-1}(\cdot) \text{ is the inverse } \Phi(\cdot).$$

We define $\mathbb{I}(\alpha) := \prod_{j=1}^{(n-1)d} \mathbb{I}_j(\alpha)$ as the Cartesian product of all $\mathbb{I}_j(\alpha)$. We then have

$$p(\alpha) = \mathbb{P}\left[\hat{\boldsymbol{t}} \notin \mathbb{I}(\alpha)\right], \text{ where } \hat{\boldsymbol{t}} \sim \mathcal{N}(\mathbf{0}, \tilde{V}).$$

We have now completed the proof of *Step 3*.

- *Step 4.* The probability $p(\alpha)$ is strictly decreasing in α with $\lim_{\alpha \downarrow 0} p(\alpha) = 0$.

It is clear by definition that $\mathbb{I}_j(\alpha_1) \subset \mathbb{I}_j(\alpha_2)$ for $\alpha_1 > \alpha_2$, so $\mathbb{I}(\alpha_1) = \prod_{j=1}^{(n-1)d} \mathbb{I}_j(\alpha_1) \subset \prod_{j=1}^{(n-1)d} \mathbb{I}_j(\alpha_2) = \mathbb{I}(\alpha_2)$ for $\alpha_1 > \alpha_2$. Therefore, for $\alpha_1 > \alpha_2$,

$$p(\alpha_1) = \mathbb{P}\left[\hat{\boldsymbol{t}} \notin \mathbb{I}(\alpha_1)\right] > \mathbb{P}\left[\hat{\boldsymbol{t}} \notin \mathbb{I}(\alpha_1)\right] = p(\alpha_2),$$

where the inequality follows from $\mathbb{I}(\alpha_1) \subset \mathbb{I}(\alpha_2)$. Finally, to prove that $\lim_{\alpha \downarrow 0} p(\alpha) = 0$, we note that $\lim_{\alpha \downarrow 0} \Phi^{-1}\left(1 - \frac{\alpha}{2}\right) = +\infty$. Thus, $\lim_{\alpha \downarrow 0} \mathbb{I}(\alpha) = \mathbb{R}^{(n-1)d}$, which implies that

$$\lim_{\alpha \downarrow 0} p(\alpha) = \lim_{\alpha \downarrow 0} \mathbb{P}\left[\hat{\boldsymbol{t}} \notin \mathbb{I}(\alpha)\right] = 0,$$

where the last equality follows from the monotone convergence theorem. This completes the proof of *Step 4*. Furthermore, by combining *Step 1*, *Step 2*, *Step 3*, and *Step 4*, we conclude that inequality (7) holds.

- *Step 5.* The DAC_α estimator $\hat{\boldsymbol{\beta}}^\alpha$ is consistent.

If the data aggregation levels $(\mathcal{D}_s, \mathcal{D}_c, \mathcal{D}_n)$ and the cluster structure $\{\mathcal{C}_{1,l}, \mathcal{C}_{2,l}, ..., \mathcal{C}_{k_l,l}\}$ are correctly identified, Proposition 2 implies the consistency of $\hat{\boldsymbol{\beta}}^{\alpha}$. We next consider the following two cases: (i) A Type-I error occurs, under which Algorithm 1 falsely identifies two identical coefficients to be different; and (ii) A Type-II error occurs, under which Algorithm 1 falsely identifies two different coefficients to be the same. *Step 1* implies that the probability of Type-II error converges to 0 as the sample size m goes to infinity. For the case of Type-I error, the model is not mis-specified and, as a consequence, the same argument as in the proof of Proposition 1(a) implies that Step 7 of Algorithm 1 consistently estimates the true coefficient $\boldsymbol{\beta}$. This completes the proof of *Step 5* and of Proposition 3(a).

Part (b). Once again, we consider the following three cases:

- *Case 1.* The data aggregation levels $(\mathcal{D}_s, \mathcal{D}_c, \mathcal{D}_n)$ and the cluster structure $\{\mathcal{C}_{1,l}, \mathcal{C}_{2,l}, ..., \mathcal{C}_{k_l,l}\}$ are correctly identified. The event of this case is denoted by $\mathcal{E}_1(m)$, where we make the dependence on the sample size m explicit.
- *Case 2.* A Type-I error occurs but there is no Type-II error, under which Algorithm 1 falsely identifies two identical coefficients to be different. The event of this case is denoted by $\mathcal{E}_2(m)$, where we make the dependence on the sample size m explicit.
- *Case 3.* A Type-II error occurs, under which Algorithm 1 falsely identifies two different coefficients to be the same. The event of this case is denoted by $\mathcal{E}_3(m)$, where we make the dependence on the sample size m explicit.

We note that $\mathbb{P}\Big[\mathcal{E}_1(m) \cup \mathcal{E}_2(m) \cup \mathcal{E}_3(m)\Big] = 1$ by definition. We first establish the following, for any $\epsilon > 0$:

$$\begin{aligned}
\mathbb{P}\Big[|\hat{\beta}^{\alpha}_{i,l}(m) - \beta_{i,l}| > \epsilon\Big] \leq & \mathbb{P}\Big[|\hat{\beta}^{\alpha}_{i,l}(m) - \beta_{i,l}| > \epsilon, \mathcal{E}_1(m)\Big] + \mathbb{P}\Big[|\hat{\beta}^{\alpha}_{i,l}(m) - \beta_{i,l}| > \epsilon, \mathcal{E}_2(m)\Big] + \mathbb{P}\Big[|\hat{\beta}^{\alpha}_{i,l}(m) - \beta_{i,l}| > \epsilon, \mathcal{E}_3(m)\Big] \\
&= \sum_{j=1}^{3} \mathbb{P}\Big[|\hat{\beta}^{\alpha}_{i,l}(m) - \beta_{i,l}| > \epsilon \Big| \mathcal{E}_j(m)\Big] \mathbb{P}\Big[\mathcal{E}_j(m)\Big] \\
&\leq \sum_{j=1}^{2} \mathbb{P}\Big[|\hat{\beta}^{\alpha}_{i,l}(m) - \beta_{i,l}| > \epsilon \Big| \mathcal{E}_j(m)\Big] + \mathbb{P}\Big[\mathcal{E}_3(m)\Big],
\end{aligned} \tag{33}$$

where the first inequality follows from the union bound and from $\mathbb{P}\Big[\mathcal{E}_1(m) \cup \mathcal{E}_2(m) \cup \mathcal{E}_3(m)\Big] = 1$, and the second inequality from $\mathbb{P}\Big[|\hat{\beta}^{\alpha}_{i,l}(m) - \beta_{i,l}| > \epsilon \Big| \mathcal{E}_2(m)\Big] \leq 1$ and $P\Big[\mathcal{E}_j(m)\Big] \leq 1$ $(j = 1, 2)$. The above equality follows from the definition of conditional probability. To prove Eq. (8), it suffices to show that there exist two constants $c_1 > 0$ and $c_2 > 0$, such that, for any $\epsilon > 0$ and sufficiently large m, the following holds:

$$\mathbb{P}\Big[|\hat{\beta}^{\alpha}_{i,l}(m) - \beta_{i,l}| > \epsilon \Big| \mathcal{E}_j(m)\Big] \leq c_1 \exp(-c_2 \epsilon^2 m), \ j = 1, 2, \tag{34}$$

$$\mathbb{P}\Big[\mathcal{E}_3(m)\Big] \leq c_1 \exp(-c_2 \epsilon^2 m). \tag{35}$$

We next quantify the concentration bounds in Eqs. (34) and (35) for the three cases separately.

Case 1. In this case, $\mathcal{E}_1(m)$ holds true. Therefore, inequality (34) (for $j = 1$) follows immediately from Proposition 2(b).

Case 2. In this case, $\mathcal{E}_2(m)$ holds true. There are $O(nd)$ sub-cases that differ on the estimation results of the data aggregation levels. For each sub-case, Proposition 2(b) also holds (though each with a different

aggregate model to estimate). Thus, by applying the law of total probability, inequality (34) (for $j=2$) follows.

Case 3. In this case, inequality (32) implies that inequality (35) holds. Plugging inequalities (34) and (35) into (33) implies that there exist constants $\eta_{i,l}^{\alpha}>0$ and $\psi_{i,l}^{\alpha}>0$ such that inequality (8) holds for any $m>\mathfrak{m}_{i,l}^{\alpha}$ by setting the threshold $\mathfrak{m}_{i,l}^{\alpha}$ sufficiently large. It thus concludes the proof of Proposition 3(b). □

Proof of Proposition 4

Part (a). By Proposition 1(b) (Eq. (5) in particular), $\sqrt{m}(\hat{b}_{i,l}(m)-\beta_{i,l})$ converges in distribution to a single-dimensional normal distribution with mean 0 and variance $\kappa_{i,l}=(\mathcal{I}_i(\boldsymbol{\beta}_i)^{-1})_{l,l}$. Thus, we have

$$\lim_{m\uparrow+\infty} m\mathbb{E}(\hat{b}_{i,l}(m)-\beta_{i,l})^2=\lim_{m\uparrow+\infty}\mathbb{E}\Big[\sqrt{m}(\hat{b}_{i,l}(m)-\beta_{i,l})\Big]^2=\kappa_{i,l},$$

namely, Eq. (9) holds for all $1\le i\le n$ and $1\le l\le d$. This completes the proof of Proposition 4(a).

Part (b). The proof relies on analyzing the log-likelihood functions of the decentralized and aggregate models carefully. Hence, we first introduce some notation. We define the (empirical average) log-likelihood of item i with the data sample $\{(Y_{i,j},\boldsymbol{X}_{i,j}):j=1,2,\ldots,m\}$ as

$$\mathfrak{L}_i(\boldsymbol{b}_i;m):=\frac{1}{m}\sum_{j=1}^{m}\log\mathcal{L}_i(\boldsymbol{\beta}_i|Y_{i,j},\boldsymbol{X}_{i,j})=\frac{1}{m}\sum_{j=1}^{m}[Y_{i,j}\boldsymbol{X}_{ij}'\boldsymbol{b}_i-H(\boldsymbol{X}_{i,j}'\boldsymbol{b}_i)],$$

where we ignore a constant independent of data for the last equality and $H'(u)=G(u)$. Thus, the decentralized estimator for item i is $\hat{\boldsymbol{b}}_i(m)=\arg\max_{\boldsymbol{b}_i}\mathfrak{L}_i(\boldsymbol{b}_i;m)$.

We also denote the gradient and Hessian of the log-likelihood function associated with item i by $\nabla\mathfrak{L}_i(\boldsymbol{b}_i;m)$ and $\nabla_2\mathfrak{L}_i(\boldsymbol{b}_i;m)$, respectively. The Fisher information matrix with respect to the decentralized model of item i is thus given by $\mathcal{I}_i(\boldsymbol{b}_i)=-\mathbb{E}[\nabla_2\mathfrak{L}_i(\boldsymbol{b}_i;1)]$, where the expectation is taken with respect to the true value of $\boldsymbol{b}_i=\boldsymbol{\beta}_i$ and the true distribution of $(Y_{i,1},\boldsymbol{X}_{i,1})$. By using the law of large numbers, we have $\lim_{m\uparrow+\infty}\nabla_2\mathfrak{L}_i(\boldsymbol{b}_i;m)=-\mathcal{I}_i(\boldsymbol{b}_i)$. Likewise, we define the log-likelihood of all the items as follows:

$$\mathfrak{L}(\boldsymbol{b};m)=\sum_{i=1}^{n}\mathfrak{L}_i(\boldsymbol{b}_i;m):=\sum_{i=1}^{n}\frac{1}{m}\sum_{j=1}^{m}\log\mathcal{L}_i(\boldsymbol{b}_i|Y_{i,j},\boldsymbol{X}_{i,j})=\sum_{i=1}^{n}\frac{1}{m}\sum_{j=1}^{m}[Y_{i,j}\boldsymbol{X}_{ij}'\boldsymbol{b}_i-H(\boldsymbol{X}_{i,j}'\boldsymbol{b}_i)].$$

Hence, the aggregate estimator is defined by $\hat{\boldsymbol{b}}^a(m)=\arg\max_{\boldsymbol{b}\in\Xi}\mathfrak{L}(\beta;m)$, where the feasible parameter set Ξ is defined as in Eq. (6). We denote the gradient and Hessian of $\mathfrak{L}(\boldsymbol{b};m)$ as $\nabla\mathfrak{L}(\boldsymbol{b};m)=\sum_{i=1}^{n}\nabla\mathfrak{L}_i(\boldsymbol{b};m)$ and $\nabla_2\mathfrak{L}(\boldsymbol{b};m)=\sum_{i=1}^{n}\nabla_2\mathfrak{L}_i(\boldsymbol{b};m)$, respectively. We are now ready to prove Proposition 4(b) in different steps.

- *Step 1.* The aggregate estimator $\hat{\boldsymbol{b}}^a(m)$ satisfies the following expected squared error:

$$\lim_{m\uparrow+\infty} m\cdot\mathbb{E}(b_{i,l}^a(m)-\beta_{i,l})^2=\left(\frac{1}{n_{i,l}}\right)^2\cdot\left(\sum_{i'\in\mathcal{C}(i,l)}\kappa_{i',l}\right),\ \text{for all } 1\le i\le n \text{ and } 1\le l\le d, \tag{36}$$

where $\kappa_{i,l}$'s are defined in Proposition 4.

Based on the decentralized estimator, $\hat{\boldsymbol{b}}(m)$, we first construct the following new estimator $\hat{\theta}_{i,l}(m)$ for each item i and feature l:

$$\hat{\theta}_{i,l}(m)=\frac{1}{n_{i,l}}\sum_{i'\in\mathcal{C}(i,l)}\hat{b}_{i',l}(m).$$

By the consistency and asymptotic normality of $\hat{\boldsymbol{b}}(m)$ (Proposition 1), we have $\hat{\theta}_{i,l}(m) \xrightarrow{p} \beta_{i,l}$, for each i and l; and

$$\begin{aligned}\lim_{m\to+\infty} m\cdot\mathbb{E}(\hat{\theta}_{i,l}(m)-\beta_{i,l})^2 &= \lim_{m\to+\infty} m\cdot\mathbb{E}\Big(\frac{1}{n_{i,l}}\sum_{i'\in\mathcal{C}(i,l)}\hat{b}_{i',l}(m)-\beta_{i,l}\Big)^2\\ &=\Big(\frac{1}{n_{i,l}}\Big)^2\cdot\lim_{m\to+\infty} m\cdot\mathbb{E}\Big(\sum_{i'\in\mathcal{C}(i,l)}(\hat{b}_{i',l}(m)-\beta_{i,l})\Big)^2\\ &=\Big(\frac{1}{n_{i,l}}\Big)^2\cdot\sum_{i'\in\mathcal{C}(i,l)}\Big[\lim_{m\to+\infty} m\cdot\mathbb{E}\Big(\hat{b}_{i',l}(m)-\beta_{i,l}\Big)^2\Big]\\ &=\Big(\frac{1}{n_{i,l}}\Big)^2\cdot\Big(\sum_{i'\in\mathcal{C}(i,l)}\kappa_{i',l}\Big),\ \text{for } i=1,2,\ldots,n,\ \text{and } l=1,2,\ldots,d,\end{aligned} \tag{37}$$

where the third equality follows from the fact that the demand of different items are independent and the last equality follows from Proposition 4(a).

We next apply the Taylor expansion of $\nabla\mathfrak{L}_i(\cdot;m)$ around the true parameter value $\boldsymbol{\beta}_i$ for each i:

$$\nabla\mathfrak{L}_i(\hat{\boldsymbol{b}}_i(m);m)=\nabla\mathfrak{L}_i(\boldsymbol{\beta}_i;m)+\nabla_2\mathfrak{L}_i(\boldsymbol{\beta}_i;m)\cdot(\hat{\boldsymbol{b}}_i(m)-\boldsymbol{\beta}_i)+o(||\hat{\boldsymbol{b}}_i(m)-\boldsymbol{\beta}_i||),$$

where $o(\cdot)$ refers to the standard "Little-o Notation" applied to each component of the vector.

Since $\hat{\boldsymbol{b}}_i(m)$ is the maximizer of $\mathfrak{L}_i(\cdot;m)$, the first-order condition applies, that is, $\nabla\mathfrak{L}_i(\hat{\boldsymbol{b}}_i(m);m)=0$. Thus, by plugging this into the Taylor expansion of $\nabla\mathfrak{L}_i(\cdot;m)$ we obtain

$$\nabla\mathfrak{L}_i(\boldsymbol{\beta}_i;m)+\nabla_2\mathfrak{L}_i(\boldsymbol{\beta}_i;m)\cdot(\hat{\boldsymbol{b}}_i(m)-\boldsymbol{\beta}_i)+o(||\hat{\boldsymbol{b}}_i(m)-\boldsymbol{\beta}_i||)=0,\ \text{for each } i. \tag{38}$$

Analogously, we apply the Taylor expansion of $\nabla\mathfrak{L}(\cdot;m)$ around the true parameter value $\boldsymbol{\beta}$:

$$\nabla\mathfrak{L}(\hat{\boldsymbol{b}}^a;m)=\nabla\mathfrak{L}(\boldsymbol{\beta};m)+\nabla_2\mathfrak{L}(\boldsymbol{\beta};m)\cdot(\hat{\boldsymbol{b}}^a(m)-\boldsymbol{\beta})+o(||\hat{\boldsymbol{b}}^a(m)-\boldsymbol{\beta}||),$$

that is,

$$\sum_{i=1}^n\nabla\mathfrak{L}_i(\hat{\boldsymbol{b}}_i^a(m);m)=\sum_{i=1}^n\nabla\mathfrak{L}_i(\boldsymbol{\beta};m)+\sum_{i=1}^n\nabla_2\mathfrak{L}_i(\boldsymbol{\beta}_i;m)\cdot(\hat{\boldsymbol{b}}_i^a(m)-\boldsymbol{\beta}_i)+o(||\hat{\boldsymbol{b}}^a(m)-\boldsymbol{\beta}||).$$

Since $\hat{\boldsymbol{b}}^a(m)$ is the maximizer of $\mathfrak{L}(\cdot;m)$ under the constraint that $\beta_{i,l}=\beta_{i',l}$ for all $i'\in\mathcal{C}(i,l)$, we have $\sum_{i'\in\mathcal{C}(i,l)}\nabla^l\mathfrak{L}_{i'}(\hat{\beta}_{i'}(m);m)=0$, where the operator ∇^l refers to the partial derivative with respect to feature l. Thus, for each item i, it follows that

$$\sum_{i'\in\mathcal{C}(i,l)}\nabla^l\mathfrak{L}_{i'}(\hat{\boldsymbol{b}}_{i'}^a(m);m)=\sum_{i'\in\mathcal{C}(i,l)}\nabla^l\mathfrak{L}_{i'}(\boldsymbol{\beta}_{i'};m)+\sum_{i'\in\mathcal{C}(i,l)}\nabla_2^l\mathfrak{L}_{i'}(\boldsymbol{\beta}_{i'};m)\cdot(\hat{\boldsymbol{b}}_{i'}^a(m)-\boldsymbol{\beta}_{i'})+o(||\hat{\boldsymbol{b}}(m)-\boldsymbol{\beta}||)=0, \tag{39}$$

where ∇_2^l is the l-th row of the Hessian.

From Eq. (38), we have for each i and each l,

$$\sum_{i'\in\mathcal{C}(i,l)}\nabla^l\mathfrak{L}_{i'}(\hat{\boldsymbol{b}}_{i'}(m);m)=\sum_{i'\in\mathcal{C}(i,l)}\nabla^l\mathfrak{L}_{i'}(\boldsymbol{\beta}_{i'};m)+\sum_{i'\in\mathcal{C}(i,l)}\nabla_2^l\mathfrak{L}_{i'}(\boldsymbol{\beta}_{i'};m)\cdot(\hat{\boldsymbol{b}}_{i'}(m)-\boldsymbol{\beta}_{i'})+o(||\hat{\boldsymbol{b}}(m)-\boldsymbol{\beta}||)=0. \tag{40}$$

We note that $\hat{b}^a_{i',l}(m) = \hat{b}^a_{i,l}(m)$ for all $i' \in \mathcal{C}(i,l)$, so that in total there are $n_{i,l} = |\mathcal{C}(i,l)|$ coefficients identical to $\hat{b}^a_{i,l}(m)$. By plugging this identity into Eq. (39) and subtracting Eq. (40), we obtain the following, for each i and each l:

$$\left| n_{i,l}\hat{b}^a_{i,l}(m) - \sum_{i' \in \mathcal{C}(i,l)} \hat{b}_{i,l}(m) \right| = o(m^{-\frac{1}{2}}), \text{ i.e., } |\hat{b}^a_{i,l}(m) - \hat{\theta}_{i,l}(m)| = o(m^{-\frac{1}{2}}), \tag{41}$$

where we used the facts that $||\hat{\boldsymbol{b}}(m) - \boldsymbol{\beta}|| = O(m^{-\frac{1}{2}})$ and $||\hat{\boldsymbol{b}}^a(m) - \boldsymbol{\beta}|| = O(m^{-\frac{1}{2}})$ (by applying the strong law of large numbers, namely, $\lim_{m\uparrow+\infty} \nabla_2^l \mathfrak{L}_i(\boldsymbol{\beta}_i; m) = -\mathcal{I}_i^l(\boldsymbol{\beta}_i)$ for any i and l). Thus, by Eq. (41),

$$\mathbb{E}\left(\hat{b}^a_{i,l}(m) - \hat{\theta}_{i,l}(m)\right)^2 = o(m^{-1}) \text{ for each } i \text{ and each } l. \tag{42}$$

For each i and each l, we have

$$\begin{aligned} & m \cdot \mathbb{E}\left(\hat{b}^a_{i,l}(m) - \beta_{i,l}\right)^2 \\ =& m \cdot \mathbb{E}\left(\hat{b}^a_{i,l}(m) - \hat{\theta}_{i,l}(m) + \hat{\theta}_{i,l}(m) - \beta_{i,l}\right)^2 \\ =& m \cdot \left[\mathbb{E}\left(\hat{b}^a_{i,l}(m) - \hat{\theta}_{i,l}(m)\right)^2 + \mathbb{E}\left(\hat{\theta}_{i,l}(m) - \beta_{i,l}\right)^2 + 2\mathbb{E}\left(\hat{b}^a_{i,l}(m) - \hat{\theta}_{i,l}(m)\right)\left(\hat{\theta}_{i,l}(m) - \beta_{i,l}\right)\right] \end{aligned} \tag{43}$$

By Eq. (37), we have

$$\lim_{\uparrow+\infty} m \cdot \mathbb{E}\left(\hat{b}^a_{i,l}(m) - \hat{\theta}_{i,l}(m)\right)^2 = \left(\frac{1}{n_{i,l}}\right)^2 \cdot \left(\sum_{i' \in \mathcal{C}(i,l)} \kappa_{i',l}\right). \tag{44}$$

By Eq. (42), we have

$$\lim_{m\uparrow+\infty} m \cdot \mathbb{E}\left(\hat{\theta}_{i,l}(m) - \beta_{i,l}\right)^2 = 0. \tag{45}$$

By Eq. (37) and Eq. (42), we have

$$\begin{aligned} & \lim_{m\uparrow+\infty} m \cdot \left| \mathbb{E}\left(\hat{b}^a_{i,l}(m) - \hat{\theta}_{i,l}(m)\right)\left(\hat{\theta}_{i,l}(m) - \beta_{i,l}\right) \right| \\ \leq & \lim_{m\uparrow+\infty} 2m \cdot \sqrt{\mathbb{E}\left(\hat{b}^a_{i,l}(m) - \hat{\theta}_{i,l}(m)\right)^2 \cdot \mathbb{E}\left(\hat{\theta}_{i,l}(m) - \beta_{i,l}\right)^2} \\ = & 2\sqrt{\left(\lim_{m\uparrow+\infty} m \cdot \mathbb{E}\left(\hat{b}^a_{i,l}(m) - \hat{\theta}_{i,l}(m)\right)^2\right) \cdot \left(\lim_{m\uparrow+\infty} m \cdot \mathbb{E}\left(\hat{\theta}_{i,l}(m) - \beta_{i,l}\right)^2\right)} \\ = & 0, \end{aligned} \tag{46}$$

where the inequality follows from the Cauchy-Schwartz inequality, the first equality from the fact that both limits exist, and the last from Eqs. (44) and (45). Finally, we plug Eqs. (44), (45), and (46) into Eq. (43) to obtain Eq. (36), and this concludes the proof of *Step 1*.

- *Step 2.* For the DAC_α estimator $\hat{\boldsymbol{\beta}}^\alpha$, inequality (10) holds.

We consider the three cases defined in the proof of Proposition 3: (i) $\mathcal{E}_1(m)$ (i.e., data aggregation levels and cluster structures corrected identified by DAC_α), (ii) $\mathcal{E}_2(m)$ (Type-I error made but no Type-II error made by DAC_α), and (iii) $\mathcal{E}_3(m)$ (Type-II error made by DAC_α).

Since $\mathbb{P}\Big[\mathcal{E}_1(m)\cup\mathcal{E}_2(m)\cup\mathcal{E}_3(m)\Big]=1$, we have

$$\begin{aligned}\mathbb{E}\Big[\hat{\beta}^{\alpha}_{i,l}(m)-\beta_{i,l}\Big]^2 \le& \mathbb{E}\Big[\Big(\hat{\beta}^{\alpha}_{i,l}(m)-\beta_{i,l}\Big)\mathbf{1}_{\mathcal{E}_1(m)}\Big]^2+\mathbb{E}\Big[\Big(\hat{\beta}^{\alpha}_{i,l}(m)-\beta_{i,l}\Big)\mathbf{1}_{\mathcal{E}_2(m)}\Big]^2+\mathbb{E}\Big[\Big(\hat{\beta}^{\alpha}_{i,l}(m)-\beta_{i,l}\Big)\mathbf{1}_{\mathcal{E}_3(m)}\Big]^2\\ =&\sum_{j=1}^{3}\mathbb{E}\Big[\Big(\hat{\beta}^{\alpha}_{i,l}(m)-\beta_{i,l}\Big)^2\Big|\mathcal{E}_j(m)\Big]\mathbb{P}\Big[\mathcal{E}_j(m)\Big]\\ \le&\mathbb{E}\Big[\Big(\hat{\beta}^{\alpha}_{i,l}(m)-\beta_{i,l}\Big)^2\Big|\mathcal{E}_1(m)\Big]\mathbb{P}\Big[\mathcal{E}_1(m)\Big]+\mathbb{E}\Big[\Big(\hat{\beta}^{\alpha}_{i,l}(m)-\beta_{i,l}\Big)^2\Big|\mathcal{E}_2(m)\Big]\mathbb{P}\Big[\mathcal{E}_2(m)\Big]+4\bar{\beta}^2\mathbb{P}\Big[\mathcal{E}_3(m)\Big],\end{aligned} \tag{47}$$

where the first inequality follows from the union bound, the second from $\Big(\hat{\beta}^{\alpha}_{i,l}(m)-\beta_{i,l}\Big)^2\le 4\bar{\beta}^2$, and the equality from the definition of conditional expectation.

We next bound each of the three terms in Eq. (47). By Proposition 3 (inequality (7) in particular), Eq. (9), Eq. (36) and, $\mathbb{P}\Big[\mathcal{E}_1(m)\Big]+\mathbb{P}\Big[\mathcal{E}_2(m)\Big]\le 1$, we obtain

$$\begin{aligned}&\lim_{m\uparrow+\infty} m\mathbb{E}\Big[\Big(\hat{\beta}^{\alpha}_{i,l}(m)-\beta_{i,l}\Big)^2\Big|\mathcal{E}_1(m)\Big]\mathbb{P}\Big[\mathcal{E}_1(m)\Big]+\lim_{m\uparrow+\infty} m\mathbb{E}\Big[\Big(\hat{\beta}^{\alpha}_{i,l}(m)-\beta_{i,l}\Big)^2\Big|\mathcal{E}_2(m)\Big]\mathbb{P}\Big[\mathcal{E}_2(m)\Big]\\ \le&(1-p(\alpha))\left(\frac{1}{n_{i,l}}\right)^2\cdot\left(\sum_{i'\in\mathcal{C}(i,l)}\kappa_{i',l}\right)+p(\alpha)\kappa_{i,l}.\end{aligned} \tag{48}$$

By invoking inequality (35) in the proof of Proposition 3, we have

$$\lim_{m\uparrow+\infty} 4\bar{\beta}^2 m\mathbb{P}\Big[\mathcal{E}_3(m)\Big]\le 4\bar{\beta}^2\cdot\lim_{m\uparrow+\infty}\big[mc_1\exp(-c_2m)\big]=0. \tag{49}$$

Plugging inequalities (48) and (49) into inequality (47) implies that inequality (10) holds. This completes the proof of *Step 2*, and, thus, the proof of Proposition 4(b).

Part (c). Since $\left(\frac{1}{n_{i,l}}\right)^2\cdot\Big(\sum_{i'\in\mathcal{C}(i,l)}\kappa_{i',l}\Big)<\kappa_{i,l}$ and $0<p(\alpha)<1$ (by Proposition 3), we have

$$(1-p(\alpha))\Big(\frac{1}{n_{i,l}}\Big)^2\cdot\Big(\sum_{i'\in\mathcal{C}(i,l)}\kappa_{i',l}\Big)+p(\alpha)\kappa_{i,l}<\kappa_{i,l}.$$

Hence, for a sufficiently large m, by Proposition 4(a) and (b), we have

$$m\cdot\mathbb{E}\Big[\hat{\beta}^{\alpha}_{i,l}(m)-\beta_{i,l}\Big]^2<m\cdot\mathbb{E}\Big[\hat{b}_{i,l}(m)-\beta_{i,l}\Big]^2,$$

which implies Eq. (11). This completes the proof of Proposition 4(c). □

Proof of Proposition 5

Part (a). Since $\lambda_{\min}(\Sigma_i)>0$ for any item i, the matrix $\boldsymbol{X}_i'\boldsymbol{X}_i$ is of full rank (i.e., rank$=d$) when the sample size m is sufficiently large. We denote the MSE of item i with respect to an estimator $\hat{\boldsymbol{\beta}}_i$ as

$$\widehat{\mathcal{MSE}}_i(\hat{\boldsymbol{\beta}}_i)=\frac{\sum_{j=1}^{m}(Y_{i,j}-\boldsymbol{X}_{i,j}\hat{\boldsymbol{\beta}}_i)^2}{m}.$$

Applying Theorem 2.2 and its proof from Rigollet (2015) to the decentralized estimator of item i $\hat{\boldsymbol{b}}_i$ implies that

$$\mathbb{E}\left[\widehat{\mathcal{MSE}}_i(\hat{\boldsymbol{\beta}}_i)\right]\le\frac{d\sigma^2}{m}. \tag{50}$$

One should also observe the identity that

$$\frac{1}{n}\sum_{i=1}^{n}\widehat{\mathcal{MSE}}_i(\hat{\boldsymbol{b}}_i)=\frac{\sum_{i=1}^{n}\sum_{j=1}^{m}(Y_{i,j}-\boldsymbol{X}_{i,j}\hat{\boldsymbol{\beta}}_i)^2}{nm}=\widehat{\mathcal{MSE}}(\hat{\boldsymbol{b}}). \tag{51}$$

Plugging Eq. (50) into the expectation of Eq. (51) yields inequality (12) and, thus, proves Proposition 5(a).

Part (b). We prove this result in different steps.

- *Step 1.* The MSE of the aggregate estimator $\hat{\boldsymbol{b}}^a$ has the following bound:

$$\mathbb{E}\left[\widehat{\mathcal{MSE}}(\hat{\boldsymbol{b}}^a)\right]\leq\frac{4d_x\sigma^2}{nm}. \tag{52}$$

Following the proof of Proposition 2, the aggregate model can be formulated as Eq. (31) with $G(u)=u$. Furthermore, the dimension of the design matrix $\tilde{\boldsymbol{X}}$ is of dimension $n\times m$ by d_x. One can easily check that $\tilde{\boldsymbol{X}}'\tilde{\boldsymbol{X}}$ has rank d_x. Thus, by Theorem 2.2 from Rigollet (2015), Eq. (52) holds. This proves *Step 1.*

- *Step 2.* Under the DAC algorithm, we denote the event that only a Type-I error occurs but a Type-II error does not occur (i.e., the algorithm may only falsely identify two identical coefficients to be different) as $\mathcal{E}_1$ and the total number of coefficients to estimate in Step 7 of Algorithm 1 is denoted as the random variable $\tilde{\mathfrak{d}}_x(\alpha)$. We then have

$$\mathbb{E}\left[\widehat{\mathcal{MSE}}(\hat{\boldsymbol{\beta}}^\alpha)\middle|\mathcal{E}_1,\tilde{\mathfrak{d}}_x(\alpha)=\mathfrak{d}_x\right]\leq\frac{4\mathfrak{d}_x\sigma^2}{nm},\ \text{for all}\ \mathfrak{d}_x=d_x,d_x+1,...,nd. \tag{53}$$

Conditioned on $\mathcal{E}_1$ and $\tilde{\mathfrak{d}}_x(\alpha)=\mathfrak{d}_x$, the DAC_α can be viewed as the aggregate estimator in a revised aggregate model. Therefore, by applying Eq. (53) to this model implies that Eq. (53) holds, and this proves *Step 2.*

- *Step 3.* Inequality (13) holds for the DAC_α estimator.

We denote $\mathcal{E}_2$ as the event where a Type-II error occurs (i.e., the DAC algorithm falsely identifies different coefficients to be identical). Hence, $\mathbb{P}[\mathcal{E}_1\cup\mathcal{E}_2]=1$. We thus have

$$\begin{aligned}
\mathbb{E}\left[\widehat{\mathcal{MSE}}(\hat{\boldsymbol{\beta}}^\alpha)\right]\leq&\mathbb{E}\left[\widehat{\mathcal{MSE}}(\hat{\boldsymbol{\beta}}^\alpha)\cdot\mathbf{1}_{\mathcal{E}_1}\right]+\mathbb{E}\left[\widehat{\mathcal{MSE}}(\hat{\boldsymbol{\beta}}^\alpha)\cdot\mathbf{1}_{\mathcal{E}_2}\right]\\
=&\mathbb{E}\left[\widehat{\mathcal{MSE}}(\hat{\boldsymbol{\beta}}^\alpha)\middle|\mathcal{E}_1\right]\mathbb{P}[\mathcal{E}_1]+\mathbb{E}\left[\widehat{\mathcal{MSE}}(\hat{\boldsymbol{\beta}}^\alpha)\middle|\mathcal{E}_2\right]\mathbb{P}[\mathcal{E}_2]\\
=&\sum_{\mathfrak{d}_x=d}^{d_x}\mathbb{E}\left[\widehat{\mathcal{MSE}}(\hat{\boldsymbol{\beta}}^\alpha)\middle|\mathcal{E}_1,\tilde{\mathfrak{d}}_x(\alpha)=\mathfrak{d}_x\right]\mathbb{P}[\mathcal{E}_1,\tilde{\mathfrak{d}}_x(\alpha)=\mathfrak{d}_x]+\mathbb{E}\left[\widehat{\mathcal{MSE}}(\hat{\boldsymbol{\beta}}^\alpha)\middle|\mathcal{E}_2\right]\mathbb{P}[\mathcal{E}_2]\\
\leq&\sum_{\mathfrak{d}_x=d}^{d_x}\frac{4\mathfrak{d}_x\sigma^2}{nm}\mathbb{P}[\tilde{\mathfrak{d}}_x(\alpha)=\mathfrak{d}_x]+4\bar{\beta}^2c_1\exp(-c_2m)\\
=&\frac{4d_x(\alpha)\sigma^2}{nm}+o(m^{-1}),
\end{aligned} \tag{54}$$

where the first inequality follows from the union bound, the second from inequality Eq. (53) and inequality (35), and the last equality from the definition of $d_x(\alpha)$ and the identity $\lim_{m\uparrow+\infty} m\exp(-c_2 m)=0$. By Eq. (54), to prove Eq. (13), it suffices to show that $d_x(\alpha)<nd$, which holds by the support of $\tilde{\mathfrak{d}}_x(\alpha)$. This completes the proof of *Step 3*.

- *Step 4*. The function $d_x(\alpha)$ is decreasing in α with $\lim_{\alpha\downarrow 0} d_x(\alpha)=d_x$.

By *Steps 3 and 4* from the proof of Proposition 3(a), Step 2 of Algorithm 1 is less likely to reject $H^l_{1,i}$ for each i and each l with a smaller α, for any sample path of $\boldsymbol{X}$ and $\boldsymbol{\epsilon}$. Thus, for any sample path of $\boldsymbol{X}$ and $\boldsymbol{\epsilon}$, $\tilde{\mathfrak{d}}_x(\alpha)$ is decreasing in α. Hence, $d_x(\alpha)=\mathbb{E}\left[\tilde{\mathfrak{d}}_x(\alpha)\right]$ is decreasing in α as well. Finally, as $\alpha\downarrow 0$, the probability that Step 2 of Algorithm 1 will reject $H^l_{1,i}$ converges to 0, which implies that $\tilde{\mathfrak{d}}_x(\alpha)$ converges to d_x with probability 1. Therefore, $\lim_{\alpha\downarrow 0} d_x(\alpha)=\lim_{\alpha\downarrow 0}\mathbb{E}\left[\tilde{\mathfrak{d}}_x(\alpha)\right]=d_x$, where the second equality follows from the monotone convergence theorem. This completes the proof of *Step 4* and of Proposition 5(b). □

Proof of Proposition 6

As a first step, we adopt the bias-variance decomposition (e.g., Eq. (7.9) in Hastie et al. 2019) to evaluate the generalization error of each item i. For any estimation algorithm π, we have

$$\begin{aligned}\mathcal{GE}_i(\pi)=&\mathbb{E}\Big[Y_{i,m_i+1}-\sum_{l\in D}\hat{\beta}_{i,l}(\pi,\mathfrak{D}_{\mathsf{tr}})X^l_{i,m_i+1}\Big]^2\\ =&\sigma^2+\mathbb{E}\Big(\sum_{l\in D}\beta_{i,l}X^l_{i,m_i+1}-\mathbb{E}\Big[\sum_{l\in D}\hat{\beta}_{i,l}(\pi,\mathfrak{D}_{\mathsf{tr}})X^l_{i,m_i+1}\Big]\Big)^2\\ &+\mathbb{E}\Big(\sum_{l\in D}\hat{\beta}_{i,l}(\pi,\mathfrak{D}_{\mathsf{tr}})X^l_{i,m_i+1}-\mathbb{E}\Big[\sum_{l\in D}\hat{\beta}_{i,l}(\pi,\mathfrak{D}_{\mathsf{tr}})X^l_{i,m_i+1}\Big]\Big)^2,\end{aligned}\tag{55}$$

where the first term is referred to as the irreducible error, the second term as the bias (which we denote as $\mathbb{B}_i(\pi)$), and the third term as the variance (which we denote as $\mathbb{V}_i(\pi)$). In the following, we will evaluate the bias and variance terms for the Dec and DAC estimators. We also use Agg to denote the aggregate estimator.

Part (a). The Dec and Agg estimators are ordinary least squares (OLS). It is a standard result in the statistics and econometrics literature that OLS is an unbiased estimator, that is, $\mathbb{B}_i(\pi)=0$ for $\pi\in\{\mathsf{Agg},\mathsf{Dec}\}$ or, equivalently,

$$\sum_{l\in D}\beta_{i,l}X^l_{i,m_i+1}-\mathbb{E}\Big[\sum_{l\in D}\hat{\beta}_{i,l}(\pi)X^l_{i,m_i+1}\Big]=0,\ \text{for } \pi\in\{\mathsf{Agg},\mathsf{Dec}\}.\tag{56}$$

We next quantify the variance term $\mathbb{V}_i(\pi)$ for item i and estimation algorithm π. For the training data of item i, we use the $m_i\times d$ matrix $\boldsymbol{X}_i(\mathsf{tr}):=(X^l_{i,j}(\mathsf{tr}):1\le l\le d,1\le j\le m)$ as the feature

matrix, and the m dimensional vector $\boldsymbol{Y}_i(\mathsf{tr}) := (Y_{i,j}(\mathsf{tr}) : 1 \le j \le m)$ as the label. For each item i, the decentralized estimator is given by:

$$\begin{aligned}\hat{\boldsymbol{\beta}}_i(\mathsf{Dec}, \mathfrak{D}_i(\mathsf{tr})) =& (\boldsymbol{X}_i(\mathsf{tr})^T X_i(\mathsf{tr}))^{-1} \boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{Y}_i(\mathsf{tr}) \\ =& (\boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{X}_i(\mathsf{tr}))^{-1} \boldsymbol{X}_i(\mathsf{tr})^T (\boldsymbol{X}_i(\mathsf{tr})\boldsymbol{\beta}_i + \boldsymbol{\epsilon}_i) \\ =& \boldsymbol{\beta}_i + (\boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{X}_i(\mathsf{tr}))^{-1} \boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{\epsilon}_i,\end{aligned} \tag{57}$$

where the first equality follows from $\boldsymbol{Y}_i(\mathsf{tr}) = \boldsymbol{X}_i(\mathsf{tr})\boldsymbol{\beta}_i + \boldsymbol{\epsilon}_i$. We are now ready to evaluate the variance of the Dec estimator:

$$\begin{aligned}\mathbb{V}_i(\mathsf{Dec}) =& \mathbb{E}\Big(\sum_{l\in D} \hat{\beta}_i(\mathsf{Dec}, \mathfrak{D}_i(\mathsf{tr})) X^l_{i,m_i+1} - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\mathsf{Dec}, \mathfrak{D}_i(\mathsf{tr})) X^l_{i,m_i+1}\Big]\Big)^2 \\ =& \mathbb{E}\Big[(\boldsymbol{X}^T_{i,m_i+1}(\boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{X}_i(\mathsf{tr}))^{-1} \boldsymbol{X}_i(\mathsf{tr})^T)\boldsymbol{\epsilon}_i \boldsymbol{\epsilon}_i^T (\boldsymbol{X}^T_{i,m_i+1}(\boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{X}_i(\mathsf{tr}))^{-1} \boldsymbol{X}_i(\mathsf{tr})^T)^T\Big] \\ =& \mathbb{E}\Big[\mathbb{E}_{\boldsymbol{\epsilon}_i}\Big[(\boldsymbol{X}^T_{i,m_i+1}(\boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{X}_i(\mathsf{tr}))^{-1} \boldsymbol{X}_i(\mathsf{tr})^T)\boldsymbol{\epsilon}_i \boldsymbol{\epsilon}_i^T (\boldsymbol{X}^T_{i,m_i+1}(\boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{X}_i(\mathsf{tr}))^{-1} \boldsymbol{X}_i(\mathsf{tr})^T)^T \Big| \boldsymbol{X}_{i,m_i+1}, \boldsymbol{X}_i(\mathsf{tr})\Big]\Big] \\ =& \sigma^2 \mathbb{E}\Big[(\boldsymbol{X}^T_{i,m_i+1}(\boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{X}_i(\mathsf{tr}))^{-1} \boldsymbol{X}_i(\mathsf{tr})^T)(\boldsymbol{X}^T_{i,m_i+1}(\boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{X}_i(\mathsf{tr}))^{-1} \boldsymbol{X}_i(\mathsf{tr})^T)^T\Big] \\ =& \sigma^2 \mathbb{E}\Big[\boldsymbol{X}^T_{i,m_i+1}(\boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{X}_i(\mathsf{tr}))^{-1} \boldsymbol{X}_{i,m_i+1}\Big] \\ =& \frac{\sigma^2}{m_i} \mathbb{E}\Big[\boldsymbol{X}^T_{i,m_i+1}\Big(\frac{1}{m_i}\boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{X}_i(\mathsf{tr})\Big)^{-1} \boldsymbol{X}_{i,m_i+1}\Big],\end{aligned}$$

where the second equality follows from Eq. (57), the third from the law of iterated expectations, and the fourth from the fact that $\epsilon_{i,j}$ are *i.i.d.* with mean 0 and variance σ^2. By the strong law of large numbers and the dominated convergence theorem, we have $\lim_{m_i\uparrow+\infty} \frac{1}{m_i}\boldsymbol{X}_i(\mathsf{tr})^T \boldsymbol{X}_i(\mathsf{tr}) = \boldsymbol{Id}_d$ and we can interchange the limit and expectation operators, where $\boldsymbol{Id}_d$ is the identity matrix with dimension d, observing that the features are *i.i.d.* with mean 0 and variance 1. Thus, we have

$$\lim_{m_i\uparrow+\infty} m_i \cdot \mathbb{V}_i(\mathsf{Dec}) = \sigma^2 \mathbb{E}\Big[\boldsymbol{X}_i^T \boldsymbol{X}_i\Big] = \sigma^2 \cdot d (= \sigma^2 \cdot (d_s + d_n + d_c)),$$

where the second equality follows from the fact that $\boldsymbol{X}_{i,m_i+1}$ has d features which are *i.i.d.* with mean 0 and variance 1. For item i, $m_i = m\tau_i$, so we obtain

$$\lim_{m\uparrow+\infty} m \cdot \mathbb{V}_i(\mathsf{Dec}) = \frac{\sigma^2 d}{\tau_i}. \tag{58}$$

Hence, by plugging Eqs. (56) and (58) into Eq. (55), we conclude that Eq. (14) holds. This proves Proposition 6(a).

Part (b). We decompose the proof of this part into several steps.

- *Step 1.* For the aggregate estimator $\hat{\boldsymbol{b}}^a$, the generalized error satisfies

$$\lim_{m\uparrow+\infty} m \cdot \big(\mathcal{GE}_i(\mathsf{Agg}) - \sigma^2\big) = \Big(\sum_{l=1}^{d} \frac{1}{\tau(i,l)}\Big) \cdot \sigma^2, \text{ for } i = 1, 2, ..., n. \tag{59}$$

Without loss of generality, we assume that the first d_s features $\{1, 2, ..., d_s\}$ are at the aggregate level, the next d_n features $\{d_s + 1, d_s + 2, ..., d_s + d_n\}$ are at the individual level, and the remaining

d_c features $\{d_n+d_s+1, d_n+d_s+2, ..., d\}$ are at the cluster level. For item i, we denote by $\boldsymbol{X}_i^s(\mathsf{tr}) = (X_{1,j}^l(\mathsf{tr}) : 1 \le l \le d_s, 1 \le j \le m_i)$ the feature matrix at the aggregate level, $\boldsymbol{X}_i^n(\mathsf{tr}) = (X_{i,j}^l(\mathsf{tr}) : d_s+1 \le l \le d_s+d_n, 1 \le j \le m_i)$ the feature matrix at the individual level, and $\boldsymbol{X}_i^c(\mathsf{tr}) = (X_{i,j}^l(\mathsf{tr}) : d_s+d_c+1 \le l \le d, 1 \le j \le m_i)$ the feature matrix at the cluster level.

We next analyze the aggregate model. The true data-generating process (of the training set) can be specified as:

$$\tilde{\boldsymbol{Y}} = \tilde{\boldsymbol{X}}(\mathsf{tr})\tilde{\boldsymbol{\beta}} + \tilde{\boldsymbol{\epsilon}},$$

where $\tilde{\boldsymbol{Y}}$ is a (τm)-dimensional label vector that concatenates $\boldsymbol{Y}_i$'s for all item i's, $\tilde{\boldsymbol{\epsilon}}$ is a (τm)-dimensional error vector that concatenates $\boldsymbol{\epsilon}_i$'s for all item i's, and $\tilde{\boldsymbol{X}}(\mathsf{tr})$ is the $(\tau m) \times d_x$-dimensional feature matrix defined as follows:

$$\tilde{\boldsymbol{X}}(\mathsf{tr}) := \begin{pmatrix} \boldsymbol{X}_1^s(\mathsf{tr}), & \boldsymbol{X}_1^n(\mathsf{tr}), & \boldsymbol{0}, & ..., & \boldsymbol{0}, & \tilde{\boldsymbol{X}}_1^c \\ \boldsymbol{X}_2^s(\mathsf{tr}), & \boldsymbol{0}, & \boldsymbol{X}_2^n(\mathsf{tr}), & ..., & \boldsymbol{0}, & \tilde{\boldsymbol{X}}_2^c \\ ..., & ..., & ..., & ..., & ..., & ... \\ \boldsymbol{X}_n^s(\mathsf{tr}), & \boldsymbol{0}, & \boldsymbol{0}, & ..., & \boldsymbol{X}_n^n(\mathsf{tr}), & \tilde{\boldsymbol{X}}_n^c \end{pmatrix},$$

where $\tilde{\boldsymbol{X}}_i^c$ ($m_i \times (\sum_{l \in \mathcal{D}_c} k_l)$-dimensional) is the design matrix block of the aggregate model with respect to item i ($i = 1, 2, ..., n$), which is constructed in a similar fashion as the aggregate-level and individual-level features. For conciseness, we do not write out $\tilde{\boldsymbol{X}}_i^c$ in detail. Thus, $\tilde{\boldsymbol{\beta}}$ is a d_x-dimensional vector where the first d_s entries are the coefficients of the features at the aggregate level, the next nd_n entries are the coefficients of the features at the individual level for each of the n items, and the last $\sum_{l \in \mathcal{D}_c} d_l$ entries are the coefficients of the features at the cluster level. As a result, for the aggregate model, the coefficient estimates are given by:

$$\hat{\tilde{\boldsymbol{\beta}}}(\mathsf{Agg}, \mathfrak{D}(\mathsf{tr})) = (\tilde{\boldsymbol{X}}(\mathsf{tr})^T \tilde{\boldsymbol{X}}(\mathsf{tr}))^{-1} \tilde{\boldsymbol{X}}(\mathsf{tr})^T \tilde{\boldsymbol{Y}}(\mathsf{tr}) = \tilde{\boldsymbol{\beta}} + (\tilde{\boldsymbol{X}}(\mathsf{tr})^T \tilde{\boldsymbol{X}}(\mathsf{tr}))^{-1} \boldsymbol{X}(\mathsf{tr})^T \tilde{\boldsymbol{\epsilon}} \in \mathbb{R}^{d_x}. \tag{60}$$

To evaluate $\mathbb{V}_i(\mathsf{Agg})$, we define an auxiliary d_x-dimensional random vector for each item i, $\tilde{\boldsymbol{X}}_{i,m_i+1}$ follows the same distribution as the $\sum_{j=1}^{i-1} m_j + 1$ row of $\tilde{\boldsymbol{X}}(\mathsf{tr})$. We also denote the non-zero entries of $\tilde{\boldsymbol{X}}_{i,m_i+1}$ as $\tilde{\mathfrak{X}}_i = ((\tilde{\boldsymbol{X}}_{i,m_i+1}^s)^T, (\tilde{\boldsymbol{X}}_{i,m_i+1}^n)^T, (\tilde{\boldsymbol{X}}_{i,m_i+1}^c)^T)^T \in \mathbb{R}^d$. We are now ready to compute $\mathbb{V}_i(\mathsf{Agg})$:

$$\begin{aligned}
&\mathbb{V}_i(\mathsf{Agg}) = \mathbb{E}\Big(\sum_{l=1}^{d_x} \hat{\tilde{\beta}}_i(\mathsf{Agg}, \mathfrak{D}_i(\mathsf{tr})) \tilde{X}_{i,m_i+1}^l - \mathbb{E}\Big[\sum_{l=1}^{d_x} \hat{\tilde{\beta}}_{i,l}(\mathsf{Agg}, \mathfrak{D}_i(\mathsf{tr})) \tilde{X}_{i,m_i+1}^l \Big] \Big)^2 \\
=&\mathbb{E}\Big[(\tilde{\boldsymbol{X}}_{i,m_i+1}^T (\tilde{\boldsymbol{X}}_i(\mathsf{tr})^T \tilde{\boldsymbol{X}}_i(\mathsf{tr}))^{-1} \tilde{\boldsymbol{X}}_i(\mathsf{tr})^T) \tilde{\boldsymbol{\epsilon}}_i \tilde{\boldsymbol{\epsilon}}_i^T (\tilde{\boldsymbol{X}}_{i,m_i+1}^T (\tilde{\boldsymbol{X}}_i(\mathsf{tr})^T \tilde{\boldsymbol{X}}_i(\mathsf{tr}))^{-1} \tilde{\boldsymbol{X}}_i(\mathsf{tr})^T)^T \Big] \\
=&\mathbb{E}\Big[\mathbb{E}_{\tilde{\boldsymbol{\epsilon}}_i}\Big[(\tilde{\boldsymbol{X}}_{i,m_i+1}^T (\tilde{\boldsymbol{X}}_i(\mathsf{tr})^T \tilde{\boldsymbol{X}}_i(\mathsf{tr}))^{-1} \tilde{\boldsymbol{X}}_i(\mathsf{tr})^T) \tilde{\boldsymbol{\epsilon}}_i \tilde{\boldsymbol{\epsilon}}_i^T (\tilde{\boldsymbol{X}}_{i,m_i+1}^T (\tilde{\boldsymbol{X}}_i(\mathsf{tr})^T \tilde{\boldsymbol{X}}_i(\mathsf{tr}))^{-1} \tilde{\boldsymbol{X}}_i(\mathsf{tr})^T)^T \Big| \tilde{\boldsymbol{X}}_{i,m_i+1}, \tilde{\boldsymbol{X}}_i(\mathsf{tr}) \Big] \Big] \\
=&\sigma^2 \mathbb{E}\Big[\tilde{\boldsymbol{X}}_{i,m_i+1}^T (\tilde{\boldsymbol{X}}_i(\mathsf{tr})^T \tilde{\boldsymbol{X}}_i(\mathsf{tr}))^{-1} \tilde{\boldsymbol{X}}_{i,m_i+1} \Big] \\
=&\frac{\sigma^2}{\tau m} \mathbb{E}\Big[\tilde{\mathfrak{X}}_i^T \Big(\frac{1}{\tau m} \widehat{\mathfrak{M}}_i(\mathsf{tr}) \Big)^{-1} \tilde{\mathfrak{X}}_i \Big],
\end{aligned} \tag{61}$$

where the second equality follows from Eq. (60), the third from the law of iterated expectations, and the fourth from the fact that $\epsilon_{i,j}$ are *i.i.d.* with mean 0 and variance σ^2. We now compute the matrix $\widehat{\mathfrak{M}}_i(\mathsf{tr})$. By the strong law of large numbers, $\widehat{\mathfrak{Z}}_i := \lim_{m\uparrow+\infty} \frac{1}{\tau m}\widehat{\mathfrak{M}}_i(\mathsf{tr})$ is a diagonal matrix with the following properties: (i) if $l \in \mathcal{D}_s$, then

$$(\widehat{\mathfrak{Z}}_i)_{l,l} = \lim_{m\uparrow+\infty} \frac{1}{\tau m} \sum_{i'=1}^{n} (\boldsymbol{X}_{i'}^l(\mathsf{tr}))^T \boldsymbol{X}_{i'}^l(\mathsf{tr}) = 1; \tag{62}$$

(ii) if $l \in \mathcal{D}_n$, then

$$(\widehat{\mathfrak{Z}}_i)_{l,l} = \frac{\tau_i}{\tau} \cdot \lim_{m\uparrow+\infty} \frac{1}{\tau_i m} (\boldsymbol{X}_{i}^l(\mathsf{tr}))^T \boldsymbol{X}_{i}^l(\mathsf{tr}) = \frac{\tau_i}{\tau}; \tag{63}$$

(iii) if $l \in \mathcal{D}_c$, then

$$(\widehat{\mathfrak{Z}}_i)_{l,l} = \frac{\tau(i,l)}{\tau} \cdot \lim_{m\uparrow+\infty} \frac{1}{\tau(i,l) m} \sum_{i' \in \mathcal{C}(i,l)} (\boldsymbol{X}_{i'}^l(\mathsf{tr}))^T \boldsymbol{X}_{i'}^l(\mathsf{tr}) = \frac{\tau(i,l)}{\tau}. \tag{64}$$

Hence, $(\widehat{\mathfrak{Z}}_i)^{-1}$ is also diagonal with $((\widehat{\mathfrak{Z}}_i)^{-1})_{l,l} = ((\widehat{\mathfrak{Z}}_i)_{l,l})^{-1}$. Therefore, by the dominated convergence theorem, we plug Eqs. (62), (63), and (64) into Eq. (61) and interchange the limit and expectation operators to obtain

$$\lim_{m\uparrow+\infty} m \cdot \mathbb{V}_i(\mathsf{Agg}) = \sigma^2 \cdot \left(\frac{d_s}{\tau} + \frac{d_n}{\tau_i} + \sum_{l \in \mathcal{D}_c} \frac{1}{\tau(i,l)} \right) = \sigma^2 \cdot \left(\sum_{l=1}^{d} \frac{1}{\tau(i,l)} \right). \tag{65}$$

Thus, by plugging Eqs. (56) and (65) into Eq. (55), we conclude that Eq. (59) holds, and this proves *Step 1*.

- *Step 2.* The generalization error of the DAC estimator satisfies Eq. (15).

We consider the three cases defined in the proof of Propositions 3 and 4: (i) $\mathcal{E}_1$ (i.e., data aggregation levels and cluster structures are correctly identified by DAC_α), (ii) $\mathcal{E}_2$ (Type-I error made but no Type-II error made by DAC_α), and (iii) $\mathcal{E}_3$ (Type-II error made by DAC_α). We define $\bar{\beta}$ as the maximum possible value of the coefficients for all items and all features.

We quantify $\mathbb{B}_i(\mathsf{DAC}_\alpha)$ and $\mathbb{V}_i(\mathsf{DAC}_\alpha)$ separately. Since $\mathbb{P}\Big[\mathcal{E}_1 \cup \mathcal{E}_2 \cup \mathcal{E}_3\Big] = 1$, we have

$$\begin{aligned}
\mathbb{B}_i(\mathsf{DAC}_\alpha) \leq & \mathbb{E}\Big[\Big(\sum_{l\in D} \beta_{i,l} X_{i,m_i+1}^l - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\mathsf{DAC}_\alpha, \mathfrak{D}_{\mathsf{tr}}) X_{i,m_i+1}^l\Big]\Big)^2 \mathbf{1}_{\mathcal{E}_1}\Big] \\
& + \mathbb{E}\Big[\Big(\sum_{l\in D} \beta_{i,l} X_{i,m_i+1}^l - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\mathsf{DAC}_\alpha, \mathfrak{D}_{\mathsf{tr}}) X_{i,m_i+1}^l\Big]\Big)^2 \mathbf{1}_{\mathcal{E}_2}\Big] \\
& + \mathbb{E}\Big[\Big(\sum_{l\in D} \beta_{i,l} X_{i,m_i+1}^l - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\mathsf{DAC}_\alpha, \mathfrak{D}_{\mathsf{tr}}) X_{i,m_i+1}^l\Big]\Big)^2 \mathbf{1}_{\mathcal{E}_3}\Big] \\
= & \sum_{j=1}^{3} \mathbb{E}\Big[\Big(\sum_{l\in D} \beta_{i,l} X_{i,m_i+1}^l - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\mathsf{DAC}_\alpha, \mathfrak{D}_{\mathsf{tr}}) X_{i,m_i+1}^l\Big]\Big)^2 | \mathcal{E}_j\Big] \cdot \mathbb{P}\Big[\mathcal{E}_j\Big] \\
\leq & 4d\bar{\beta}^2 \mathbb{P}\Big[\mathcal{E}_3\Big],
\end{aligned} \tag{66}$$

where the first inequality follows from the union bound, the second from the fact that OLS is unbiased for a correctly specified model and the definition of $\bar{\beta}$, and the equality from the definition of conditional expectation. Therefore, by inequality (35) in the proof of Proposition 3 and inequality (66) , we have

$$\lim_{m\uparrow+\infty} \mathbb{B}_i(\mathsf{DAC}_\alpha) \leq \lim_{m\uparrow+\infty} 4d\bar{\beta}^2 m\mathbb{P}\left[\mathcal{E}_3\right] \leq 4d\bar{\beta}^2 \cdot \lim_{m\uparrow+\infty} \left[mc_1 \exp(-c_2 m)\right] = 0. \tag{67}$$

We next evaluate $\mathbb{V}_i(\mathsf{DAC}_\alpha)$ using a similar strategy:

$$\begin{aligned}
\mathbb{V}_i(\mathsf{DAC}_\alpha) \leq & \mathbb{E}\Big[\Big(\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1} - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1}\Big]\Big)^2 \mathbf{1}_{\mathcal{E}_1}\Big] \\
& + \mathbb{E}\Big[\Big(\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1} - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1}\Big]\Big)^2 \mathbf{1}_{\mathcal{E}_2}\Big] \\
& + \mathbb{E}\Big[\Big(\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1} - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1}\Big]\Big)^2 \mathbf{1}_{\mathcal{E}_3}\Big] \\
= & \sum_{j=1}^{3} \mathbb{E}\Big[\Big(\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1} - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1}\Big]\Big)^2 \Big| \mathcal{E}_j\Big] \cdot \mathbb{P}\Big[\mathcal{E}_j\Big] \\
\leq & \mathbb{E}\Big[\Big(\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1} - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1}\Big]\Big)^2 \Big| \mathcal{E}_1\Big] \cdot \mathbb{P}\Big[\mathcal{E}_1\Big] \\
& + \mathbb{E}\Big[\Big(\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1} - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1}\Big]\Big)^2 \Big| \mathcal{E}_2\Big] \cdot \mathbb{P}\Big[\mathcal{E}_2\Big] \\
& + 4d\bar{\beta}^2 \mathbb{P}\Big[\mathcal{E}_3\Big],
\end{aligned} \tag{68}$$

where the first inequality follows from the union bound, the second from the definition of $\bar{\beta}$, and the equality from the definition of conditional expectation.

We now bound each of the three terms in Eq. (68). By Proposition 3 (inequality (7) in particular), Eqs. (58), (65) and, $\mathbb{P}\left[\mathcal{E}_1\right] + \mathbb{P}\left[\mathcal{E}_2\right] \leq 1$, we have

$$\begin{aligned}
& \lim_{m\uparrow+\infty} m \cdot \mathbb{E}\Big[\Big(\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1} - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1}\Big]\Big)^2 \Big| \mathcal{E}_1\Big] \cdot \mathbb{P}\Big[\mathcal{E}_1\Big] \\
& + \lim_{m\uparrow+\infty} m \cdot \mathbb{E}\Big[\Big(\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1} - \mathbb{E}\Big[\sum_{l\in D} \hat{\beta}_{i,l}(\pi, \mathfrak{D}_{\mathsf{tr}}) X^l_{i,m_i+1}\Big]\Big)^2 \Big| \mathcal{E}_2\Big] \cdot \mathbb{P}\Big[\mathcal{E}_2\Big] \\
\leq & p(\alpha) \cdot \frac{d \cdot \sigma^2}{\tau_i} + (1 - p(\alpha)) \cdot \Big(\sum_{l=1}^{d} \frac{1}{\tau(i,l)}\Big) \cdot \sigma^2.
\end{aligned} \tag{69}$$

Plugging inequalities (69) and (67) into inequality (68) implies that

$$\lim_{m\uparrow+\infty} m \cdot \mathbb{V}_i(\mathsf{DAC}_\alpha) \leq p(\alpha) \cdot \frac{d \cdot \sigma^2}{\tau_i} + (1 - p(\alpha)) \cdot \Big(\sum_{l=1}^{d} \frac{1}{\tau(i,l)}\Big) \cdot \sigma^2. \tag{70}$$

By combining inequalities (67) and (70) with the bias-variance decomposition from Eq. (55), we conclude that inequality (15) holds. This completes the proof of *Step 2*, and, thus, the proof of Proposition 6(b).

Part (c). By subtracting Eq. (14) from Eq. (15), we have

$$\lim_{m\uparrow+\infty} m\cdot\Big(\mathcal{GE}_i(\mathsf{Dec})-\mathcal{GE}_i(\mathsf{DAC}_\alpha)\Big)>(1-p(\alpha))\cdot\Big(d_s\Big(\frac{1}{\tau_i}-\frac{1}{\tau}\Big)+\sum_{l\in\mathcal{D}_c}\Big(\frac{1}{\tau_i}-\frac{1}{\tau(i,l)}\Big)\Big)\cdot\sigma^2,$$

where the inequality follows from the fact that, if a Type-I error occurs, it may still identify some (if not all) of the identical coefficients. Since $0<p(\alpha)<1$ (by Proposition 3), $\tau>\tau_i$, and $\tau(i,l)\geq\tau_i$ for each i and l, for a sufficiently large m, we have

$$m\cdot\Big(\mathcal{GE}_i(\mathsf{Dec})-\mathcal{GE}_i(\mathsf{DAC}_\alpha)\Big)>(1-p(\alpha))\cdot\Big(d_s\left(\frac{1}{\tau_i}-\frac{1}{\tau}\right)+\sum_{l\in\mathcal{D}_c}\Big(\frac{1}{\tau_i}-\frac{1}{\tau(i,l)}\Big)\Big)\cdot\sigma^2>0,$$

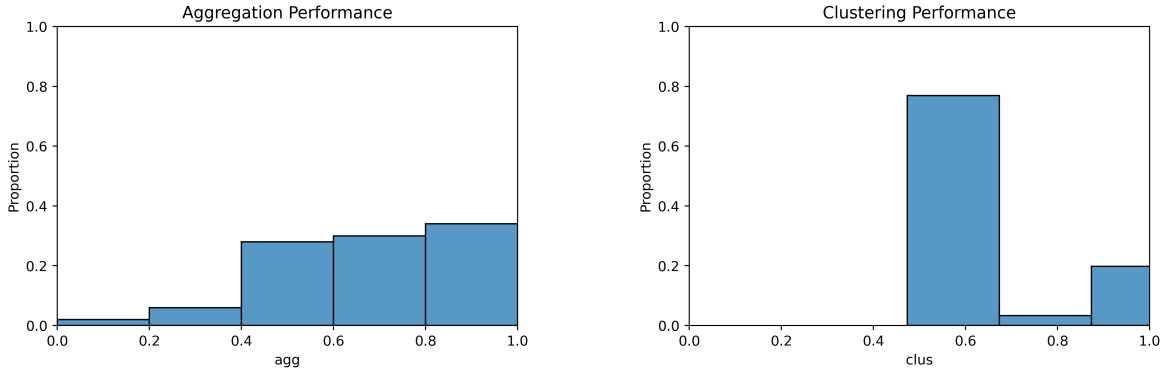
namely, Eq. (16) holds. Finally, by taking the first- and second-order partial derivatives of $g_i(\boldsymbol{\tau})$ we obtain, for $i'\neq i$,

$$\frac{\partial g_i(\boldsymbol{\tau})}{\partial\tau_i}<0,\ \frac{\partial g_i(\boldsymbol{\tau})}{\partial\tau_{i'}}>0,\ \frac{\partial^2 g_i(\boldsymbol{\tau})}{\partial_2\tau_i}>0,\ \text{and}\ \frac{\partial^2 g_i(\boldsymbol{\tau})}{\partial_2\tau_{i'}}<0.$$

This suggests that $g_i(\boldsymbol{\tau})$ is convexly decreasing in τ_i and concavely increasing in $\tau_{i'}$. We have thus completed the proof of Proposition 6(c). □

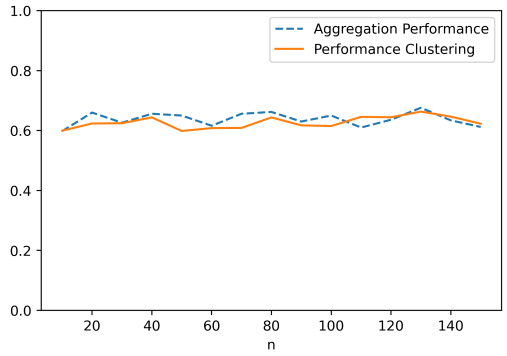
Appendix D: Additional Plots for Section 5

In this section, we report the simulation results to evaluate the performance of the DAC algorithm to identify the data aggregation levels of the features and recover the cluster structure of the items with respect to cluster-level features. The simulation details are presented in Section 5.1.

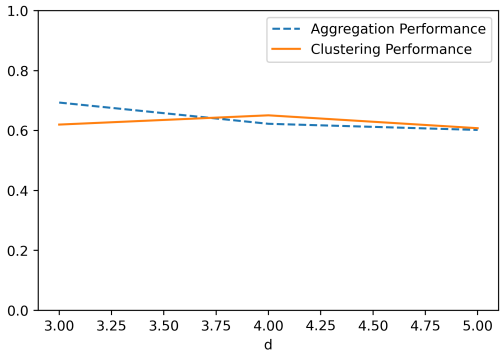


(a) Identifying aggregation levels (b) Recovering the cluster Structure

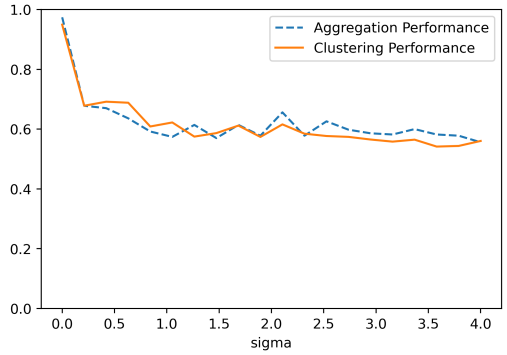
Figure 7 Performance in identifying aggregation levels and recovering the cluster structure.



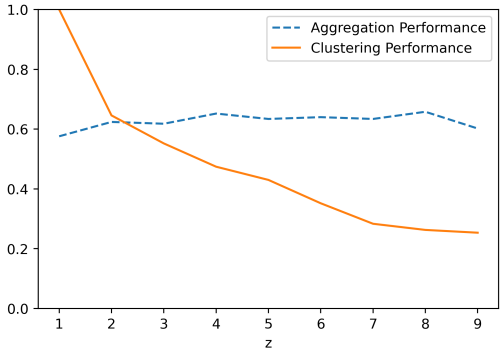
(a) Varying the number of items n



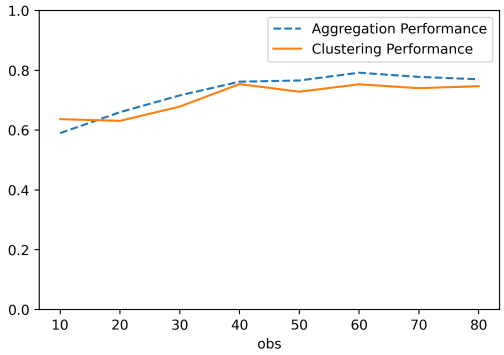
(b) Varying the number of features d



(c) Varying the noise magnitude σ

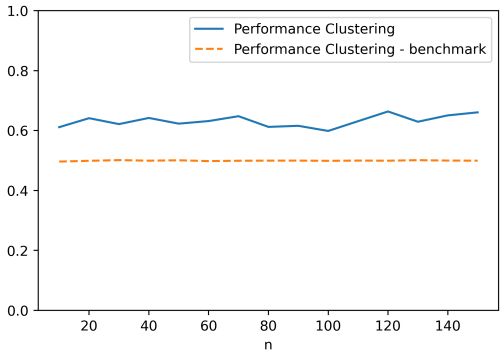


(d) Varying the number of clusters k

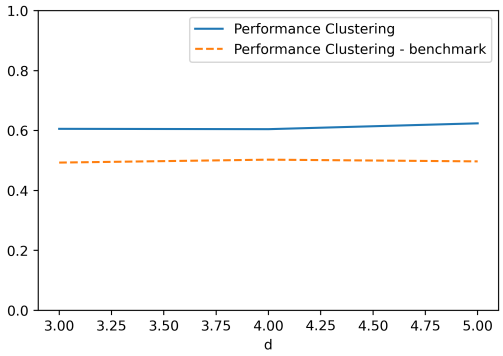


(e) Varying the number of observations m

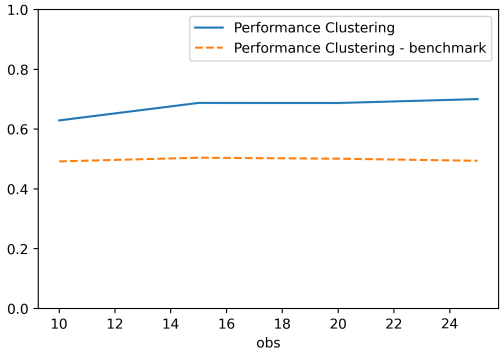
Figure 8 Sensitivity analysis on the DAC performance.



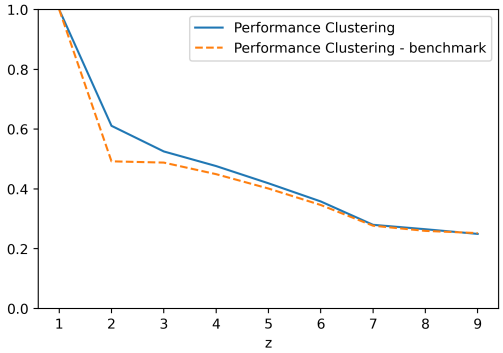
(a) Varying the number of items n



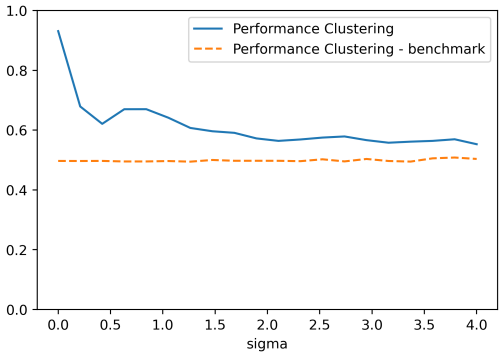
(b) Varying the number of features d



(c) Varying the number of observations m



(d) Varying the number of clusters k



(e) Varying the noise magnitude σ

Figure 9 **Comparing DAC with the k-means benchmark.**